Proceedings of the XXIV Workshop of Physical Agents

Miguel Cazorla Francisco Gomez-Donoso Felix Escalona 2024



UNIVERSIDAD DE ALICA UNIVERSITAT D'ALACAN I.U. INVESTIGACIÓN INFORMÁTIC I.U. D'INVESTIGACIÓ INFORMÁTIC



Universitat d'Alacant Universidad de Alicante







Workshop of Physical Agents 2024

PROCEEDINGS OF THE XXIV WORKSHOP OF PHYSICAL AGENTS

September 5-6, 2024

Editors:

Miguel Cazorla Francisco Gomez-Donoso Felix Escalona



2024

Proceedings of the XXIV Workshop of Physical Agents (2024)

Editors: Miguel Cazorla, Francisco Gomez-Donoso, Felix Escalona **ISBN:** 978-84-09-63822-2

Sponsored by Conselleria de Innovación, Universidades, Ciencia y Sociedad Digital de la Generalitat Valenciana and Instituto Universitario de Investigación Informática

This document is licensed under a CC BY-NC-SA International License (Creative Commons Attribution NonCommercial-ShareAlike 4.0 International License). You are free to share — copy and redistribute the material in any medium or format Adapt — remix, transform, and build upon the material. The licensor cannot revoke these freedoms as long as you follow the license terms. Under the following terms: Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. Non Commercial — You may not use the material for commercial purposes. Share Alike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

PREFACE

Welcome to the proceedings of the "Workshop of Physical Agents", a distinguished gathering that brings together researchers and practitioners from diverse fields such as computer vision, robotics, cognitive architectures, and machine learning. This workshop serves as a vibrant platform for the exchange of innovative ideas, cutting-edge research, and transformative technologies that are shaping the future of physical agents.

Physical agents, encompassing autonomous robots, intelligent systems, and adaptive software agents, are becoming increasingly integral to various aspects of our lives. From enhancing industrial automation to advancing medical diagnostics, these agents are pushing the boundaries of what is possible in both the digital and physical realms. The interdisciplinary nature of this workshop reflects the multifaceted challenges and opportunities in developing, deploying, and understanding these sophisticated systems.

This collection of scientific papers represents the latest advancements in the field. Our contributors delve into a wide array of topics, showcasing the breadth and depth of current research. The papers are organized into several key themes: Computer Vision, Intelligent Robotics, Machine learning, Social Robotics, Intelligent Robotics and Cognitive robotics.

The diversity of topics covered in this workshop underscores the collaborative spirit and interdisciplinary approach essential for advancing the field of physical agents. Each paper has undergone a rigorous peer-review process, ensuring the highest standards of scientific quality and relevance.

We extend our heartfelt gratitude to the authors for their valuable contributions, the reviewers for their insightful feedback, and the organizing committee for their dedication and hard work in making this workshop a success. We also thank our sponsors and supporters for their generous contributions, which have been instrumental in facilitating this event.

As you explore these proceedings, we hope you find inspiration and new perspectives that will drive your research and innovation in physical agents. May this workshop foster fruitful discussions, collaborations, and advancements that will shape the future of technology and its impact on society.

Welcome to the "Workshop of Physical Agents" 2024.

Sincerely,

Miguel Cazorla, Francisco Gomez-Donoso, Felix Escalona

HOST COMMITEE

Miguel Cazorla Francisco Gomez-Donoso Felix Escalona Diego Viejo Ester Martínez Martín Álvaro Belmonte Baeza Bessie Domínguez Carlos Zambrana Héctor Penadés Luis Márquez Carpintero Mónica Pina Navarro Sergio Suescun Francisco Morillas Espejo Alberto Joaquin López Sellers Germán González Serrano

PROGRAM COMMITEE

Adrián Romero Garcés. Universidad de Málaga, Spain Antonio Bandera Rubio. Universidad de Málaga, Spain Antonio Fernández Caballero. Universidad de Castilla-La Mancha, Spain Antonio González Muñoz. Universidad de Granada, Spain Carlos Vázquez Regueiro. Universidad de A Coruña, Spain Cristina Urdiales García. Universidad de Málaga, Spain Cristina Romero González. Universidad de Castilla-La Mancha, Spain Domenec Puig Valls. Universitat Rovira i Virgili, Spain Eduardo Zalama Casanova. Universidad de Valladolid, Spain Félix Escalona Moncholí. Universidad de Alicante, Spain Eugenio Aguirre Molina. Universidad de Granada, Spain Fernando Fernández Rebollo. Universidad Carlos III de Madrid, Spain Francisco Gómez Donoso. Universidad de Alicante, Spain Francisco Martín Rico. Universidad Rey Juan Carlos, Spain Humberto Martínez Barbera. Universidad de Murcia, Spain Ismael García Varea. Universidad de Castilla-La Mancha, Spain Joaquín López Fernández. Universidad de Vigo, Spain José D. Hernández Sosa. Universidad de Las Palmas de Gran Canaria, Spain José María Armingol Moreno. Universidad Carlos III de Madrid, Spain José María Cañas Plaza. Universidad Rey Juan Carlos, Spain

José Vicente Soler Bayona. Universidad Politécnica de Valencia, Spain Juan Pedro Bandera Rubio. Universidad de Málaga, Spain Lluís Ribas i Xirgo. Universidad Autónoma de Barcelona, Spain Luis Miguel Bergasa Pascual. Universidad de Alcalá de Henares, Spain Luis V. Calderita Estévez. Universidad de Málaga, Spain Manuel Mucientes Molina. Universidad de Santiago de Compostela, Spain Manuel Ocaña Miguel. Universidad de Alcalá de Henares, Spain Miguel Cazorla Quevedo. Universidad de Alicante, Spain Miguel Ángel García García. Universidad Autónoma de Madrid, Spain Miguel Garcia Silvente. Universidad de Granada, Spain Pablo Bustos García de Castro. Universidad de Extremadura, Spain Pedro Núñez Trujillo. Universidad de Extremadura, Spain Pilar Bachiller Burgos. Universidad de Extremadura, Spain Rafael Barea Navarro. Universidad de Alcalá de Henares, Spain Rafael Muñoz Salinas. Universidad de Córdoba, Spain Rebeca Marfil Robles. Universidad de Málaga, Spain Roberto Iglesias Rodríguez. Universidad de Santiago de Compostela, Spain Vicente Matellán Olivera. Universidad de León, Spain Víctor González Castro. Universidad de León, Spain

Slot	September 5, 2024 (Thursday)	September 6, 2024 (Friday)	
8:45	Credentials Pick-up		
9:15	Welcome Session		
9:30	Session 1 – Computer Vision	Session 5 – Intelligent Robots II	
10:30	Coffee Break		
11:00	Plenary I - Fundamental Challenges in	Session 6 – Cognitive Robots I	
	Deploying Autonomous Systems in Industrial and Urban Environments. Eduardo Nebot.		
12:00	Session 2 – Intelligent Robotics I	Session 7 – Cognitive Robots II	
13:00	Lunch Break		
14:00	Plenary II - Spotting robots: toward scene understanding and dynamic object		
	interactions. Zuria Bauer.		
14:30	Session 3 – Machine Learning		
15:30	Coffee Break		
16:00	Session 4 – Social Robotics		
17:00	REDAF Meeting		
18:00	Social – Santa Barbara Castle		
20:30	Dinner – Restaurante Poniente		

Session 1 - Computer Vision. Chairman: Francisco Gómez Donoso

Imitation Learning for vision based Autonomous Driving with Ackermann cars. Alejandro Moncalvillo, José-María Cañas-Plaza, Sergio Paniego, Roberto Calvo and Abdulla Al-Kaff

Upgrading the cognitive architecture of a robot to become socially aware. Jose Galeas, Pedro Núñez, Alberto Tudela, Antonio Jerez, Óscar Pons and Juan Pedro Bandera Rubio

Integration of a Sign Language Interface in a Socially Assistive Robot. Sohaila Chtioui Bouhou, Alberto J. Tudela, José Galeas, Óscar Pons, Antonio Bandera and Adrián Romero-Garcés.

Session 2 – Intelligent Robotics I. Chairman: Félix Escalona

Defining a Standard Blockchain-based Black Box for Autonomous Systems. Laura Inyesto-Alonso, David Sobrín-Hidalgo, Claudia Álvarez-Aparicio, Adrián Campazas-Vega, Ángel M. Guerrero-Higueras and Vicente Matellán-Olivera.

Formalizing a Social Navigation Use Case in PDDL. Camino Rodríguez, Alba Gragera, José Galeas, Pablo Bustos, Ángel García-Olaya and Fernando Fernández

Quadrupeds Robots in Herding: Metrics for Experimental Validation of Animal-Robot Interactions. Beatriz Jové, Alexis Gutiérrez, Camino Fernández-Llamas, Lidia Sánchez-González, Francisco Javier Rodríguez-Lera and Vicente Matellán Olivera

Session 3 – Machine Learning. Chairman: Vicente Matellán

People detection on 2D laser range finder data using deep learning and machine learning. José Abrego-González, Eugenio Aguirre and Miguel García-Silvente.

Discrete Active Inference with Dynamic Scene Graphs. Pablo Lanillos and Pablo Bustos. Analysis of strategies to apply federated learning in the field of robotics. Roi Martínez Enríquez, Nicolás Fraga Corredoira, José Miguel Burés Amatriain, Roberto Iglesias Rodríguez, Francisco Javier García Polo and Xosé Ramón Fernández Vidal

Session 4 - Social Robotics. Chairman: José María Cañas

PAWS: Personal Assistance Walking System for the Visually Impaired. Carmen Losantos-Pulido, Francisco Gomez-Donoso, Felix Escalona and Miguel Cazorla.

Re-engineering EBO: Advancing Social Robotics for Enhanced Care. Sergio Eslava Velasco, Alejandro Torrejón Harto and Pedro Nuñez Trujillo.

Multimodal Emotional Recognition for Human-Robot Interaction in Social Robotics. Sergio García-Muñoz, Sergio Suescun-Ferrandiz, Francisco Gomez-Donoso and Miguel Cazorla.

Session 5 – Intelligent Robotics II. Chairman: Pablo Bustos

Enhanced Robot Navigation in a Cyber-Physical System-of-Systems Architecture. Alberto J. Tudela, José Galeas, Óscar Pons, Juan Pedro Bandera and Antonio Bandera.

Lightweight Automated Vehicle Simulator for Studying Drivers Reactions in Real Scenarios. Mohammed Bendahmane and Lluís Ribas-Xirgo

Session 6 - Cognitive Robotics I. Chairman: Ismael García

CORTEX as an Enabler of Multi-platform Integration in Internet of Robotics Things ecosystems. Alberto J. Tudela, José Galeas, Óscar Pons, Carlos Madrid, Sara Aguilera, Pablo Gómez, Juan Pedro Bandera, Antonio Bandera and Pablo Bustos

Impact of Real time on Active Perception Systems applied to Social Robotics. Antonio Alberto García Gómez-Jacinto, Juan Diego Peña Narvaez, Rodrigo Pérez Rodríguez, José Miguel Guerrero, Francisco Martín Rico and Juan Carlos Manzanares Serrano

Enhancing Cognitive Therapies for Older Adults with EBO: An Autonomous Social Robot System Integrating LLMs and Therapist-in-the-Loop. Antonio Blanco Castillo, Jackie Lee Chong Ojeda, Alicia Condón Pérez and Pedro Miguel Núñez Trujillo.

Session 7 - Cognitive Robotics II. Chairman: Franciso Martín Rico

Graph-based inner state prediction for a socially aware robot. Jose Tejón Moreno, Juan Pedro Bandera, Antonio Bandera and Adrián Romero-Garcés

On the use of ontologies as knowledge source for social robots the INSIGHT research project. Jesus Martinez-Gomez, Ismael García-Varea, M. Julia Flores, Luis de la Ossa, Javier Ballesteros-Jerez, Antonio Bandera, Rebeca Marfil, Pilar Bachiller and Pablo Bustos

Contents

Plenary I: Fundamental Challenges in Deploying Autonomous Systems in	
Industrial and Urban Environments.	
Eduardo Nebot. Australian Centre for Robotics. University of Sydney.	1
Plenary II: Spotting Robots: Toward Scene Understanding and Dynamic	
Object Interactions	
Zuria Bauer. Computer Vision and Geometry Group. ETH Zürich.	2
PAWS: Personal Assistance Walking System for the Visually Impaired. Carmen Losantos-Pulido, Francisco Gomez-Donoso, Felix Escalona and Miguel	
Cazorla	3
Re-engineering EBO: Advancing Social Robotics for Enhanced Care.	
Sergio Eslava Velasco, Alejandro Torrejón Harto and Pedro Nuñez Trujillo	16
Enhancing Cognitive Therapies for Older Adults with EBO: An Au- tonomous Social Robot System Integrating LLMs and Therapist-in-	
the-Loop.	
Antonio Blanco Castillo, Jackie Lee Chong Ojeda, Alicia Condón Pérez and Pedro Miguel Núñez Trujillo	31
Defining a Standard Blockchain-based Black Box for Autonomous Sys- tems.	
Laura Inyesto-Alonso, David Sobrín-Hidalgo, Claudia Álvarez-Aparicio, Adrián	
Campazas-Vega, Ángel M. Guerrero-Higueras and Vicente Matellán-Olivera	46
Upgrading the cognitive architecture of a robot to become socially aware.	
Pedro Bandera Rubio	60
Formalizing a Social Navigation Use Case in PDDL.	

Camino Rodríguez, Alba Gragera, José Galeas, Pablo Bustos, Ángel García-Olaya

and Fernando Fernández

Quadrupeds Robots in Herding: Metrics for Experimental Validation of Animal-Robot Interactions. Beatriz Jové, Alexis Gutiérrez, Camino Fernández-Llamas, Lidia Sánchez-González Francisco Javier Rodríguez-Lera and Vicente Matellán Olivera	z, 89
Graph-based inner state prediction for a socially aware robot. Jose Tejón Moreno, Juan Pedro Bandera, Antonio Bandera and Adrián Romero- Garcés	103
On the use of ontologies as knowledge source for social robots the IN- SIGHT research project. Jesus Martinez-Gomez, Ismael García-Varea, M. Julia Flores, Luis de la Ossa, Javier Ballesteros-Jerez, Antonio Bandera, Rebeca Marfil, Pilar Bachiller and Pablo Bustos	116
Enhanced Robot Navigation in a Cyber-Physical System-of-Systems Ar- chitecture. Alberto J. Tudela, José Galeas, Óscar Pons, Juan Pedro Bandera and Antonio Bandera	129
CORTEX as an Enabler of Multi-platform Integration in Internet of Robotics Things ecosystems. Alberto J. Tudela, José Galeas, Óscar Pons, Carlos Madrid, Sara Aguilera, Pablo Gómez, Juan Pedro Bandera, Antonio Bandera and Pablo Bustos	143
Integration of a Sign Language Interface in a Socially Assistive Robot. Sohaila Chtioui Bouhou, Alberto J. Tudela, José Galeas, Óscar Pons, Antonio Bandera and Adrián Romero-Garcés	158
Impact of Real time on Active Perception Systems applied to Social Robotics. Antonio Alberto García Gómez-Jacinto, Juan Diego Peña Narvaez, Rodrigo Pérez Rodríguez, José Miguel Guerrero, Francisco Martín Rico and Juan Carlos Man- zanares Serrano	169
Discrete Active Inference with Dynamic Scene Graphs. Pablo Lanillos and Pablo Bustos	186
Imitation Learning for vision based Autonomous Driving with Ackermann cars. Alejandro Moncalvillo, José-María Cañas-Plaza, Sergio Paniego, Roberto Calvo and Abdulla Al-Kaff	206

Multimodal Emotional Recognition for Human-Robot Interaction in So- cial Robotics				
Sergio García-Muñoz, Francisco Gomez-Donoso and Miguel Cazorla	220			
People detection on 2D laser range finder data using deep learning and machine learning.	1			
José Abrego-González, Eugenio Aguirre and Miguel García-Silvente	235			
Lightweight Automated Vehicle Simulator for Studying Drivers Reactions in Real Scenarios.	3			
Mohammed Bendahmane and Lluís Ribas-Xirgo	250			
Analysis of strategies to apply federated learning in the field of robotics Roi Martínez Enríquez, Nicolás Fraga Corredoira, José Miguel Burés Amatri	•			

Roi Martínez Enríquez, Nicolás Fraga Corredoira, José Miguel Burés Amatriain, Roberto Iglesias Rodríguez, Francisco Javier García Polo and Xosé Ramón Fernández Vidal 265

Plenary I: Fundamental Challenges in Deploying Autonomous Systems in Industrial and Urban Environments.

In recent years, the fields of robotics and autonomous driving have witnessed rapid advancements, driven by a combination of sophisticated algorithms, improved sensor technologies, and increased computational power. However, despite these advancements, there remain significant challenges that must be addressed to achieve fully autonomous systems capable of navigating complex environments safely and efficiently. Among these challenges, precise sensor calibration, real-time motion estimation, and robust object detection are paramount.

Addressing these challenges requires a multidisciplinary approach that combines computer vision, machine learning, sensor fusion, and robotics. The development of robust solutions to these problems is crucial for the advancement of autonomous systems that can operate safely and effectively in the real world. As research continues to evolve, the pursuit of these objectives will pave the way for the next generation of autonomous technologies, bringing us closer to a future where robots and vehicles seamlessly integrate into our daily lives.

Eduardo Nebot. Australian Centre for Robotics. University of Sydney.

Plenary II: Spotting Robots: Toward Scene Understanding and Dynamic Object Interactions.

Robotic scene understanding involves the ability of robots to interpret and analyse their surroundings, enabling them to navigate and interact with real-world environments effectively. This includes recognizing and responding to dynamic object interactions, which are crucial for tasks like human-robot collaboration. This presentation explores the latest advancements in robotic scene understanding and dynamic object interactions, notably using the Spot robot. We delve into innovative approaches that enable Spot to perform complex tasks such as opening drawers and grasping objects through deep learning frameworks. Additionally, we examine the development of dynamic scene graphs that store and enrich information, creating detailed and interconnected representations of environments. The talk also covers how interactions with light switches can be integrated into these scene graphs, leading to lightweight data structures that adapt to real-time changes. Furthermore, we introduce a robotic benchmark dataset designed to enhance visual localization techniques, providing a standardized platform for evaluation and improvement. These advancements collectively aim to push the boundaries of robotics, fostering more sophisticated and dynamic interactions with the environment.

Zuria Bauer. Computer Vision and Geometry Group. ETH Zürich.

PAWS: Personal Assistance Walking System for the Visually Impaired

Carmen Losantos-Pulido¹, Francisco Gomez-Donoso¹^[0000-0002-7830-2661], Felix Escalona¹^[0000-0003-2245-601X], and Miguel Cazorla¹^[0000-0001-6805-3633]

> University Institute for Computing Research. Universidad de Alicante, 03690, Alicante, Spain. clp44@alu.ua.es, {fgomez, felix.escalona, miguel.cazorla}@ua.com

Abstract. Blindness and vision impairment are among the most common disabilities worldwide. The World Health Organization estimates that 40 to 45 million people are blind, and 135 million have low vision. There are various assistive devices designed to enhance the quality of life for visually impaired individuals. However, there is a notable gap in assistive devices specifically for navigation. Blind individuals often rely on human or animal companions to navigate, which presents its own set of issues. To address these challenges, we propose PAWS, a Personal Assistance Walking System for the visually impaired. PAWS is a robotic quadruped equipped with vision and navigation algorithms, designed to guide a blind person to specific locations in outdoor and urban environments. The base robot is a commercially available model that we modified with new hardware and software tailored to this purpose. We successfully tested PAWS on our university campus, demonstrating its potential as an effective navigation aid for the visually impaired.

Keywords: Robotics · Quadrupeds · Navigation · Visually impaired

1 Introduction

Blindness and vision impairment is one of the most common disabilities globally. In fact, an estimated 40 to 45 million people are blind and 135 million have low vision, as stated by the World Health Organization [8]. In the same document, it is also said that the 80% of the cases could be addressed to some extent to improve their vision. However, there still are a great deal of individuals that could not. The daily life for this kind of persons pose a plethora of challenges.

Nowadays, there are a range of assistive devices aimed to improve their quality of life hugely. For instance, there are digital colorimeters that read the color of any item out loud; smartphone apps to describe scenes and objects; and optical character recognition devices to understand written text. Nonetheless, there kind of assistive devices have almost no presence for tackling the navigation issue. Blind people are on their own for going to places, or they usually resolve to a human or animal companions. This pose problems as well, as it will not be a human companion available at any time. Guide dogs, on the other hand, are hard to train, very costly and, sometimes, unpredictable.

To address these matters, we propose PAWS, a Personal Assistance Walking System for the visually impaired. This is a robotic quadruped packed with vision and navigation algorithms, specially tailored to guide a blind person to an specific place in outdoors and urban environments. The robot used as base is a commercial model that was modify with new hardware and the corresponding software. We tested it in our University campus with high success.

The rest of the paper is structured as follows. First, a review of related works is provided in Section 2. Then, Section 3, describes the system and its components. In Section 4, we show the experiments we carried out to validate our approach. Finally, the conclusions and future works are drawn in Section 5.

2 Related Works

Assistive devices for the blind and vision impaired is a very interesting area of research. Being able to augment, replace, or complement human abilities, specially if they were harmed due to accidents or illnesses, is of high important from the facet of integration, quality of life and health care. Due to this, a range of approaches emerged.

They can be split in three categories: wearable devices, dedicated devices and robotic frameworks.

Another methodology is to integrate the assistance devices within the body of the user. This is the case of [1], in which an artificial-intelligence-based sonarlike system is proposed. The user wears a monocular camera that is converted to a tridimensional representation of the obstacles present in the scene. A similar approach is followed in [11], as it proposes a sonar based device with tactile feedback developed to improve the mobility and independence of visually impaired individuals. It features a transceiver/receiver, a potentiometer, a microcontroller, a rechargeable polymer lithium ion battery, and a Nokia Cell phone vibrator. Additionally, the device features an adjustable detection scheme for user customization.

A different approach would be to introduce beacons in the environment that be read with a devices. This case is studied in [10]. They propose a wireless network of "crumbs" providing access from any point in a building to a central server that provides orientation and wayfinding information. These information is extracted using a dedicated device. A similar methodology is explored in [13], in which the authors propose a system that uses RFID and GPS based localization while operating indoor and outdoor respectively. The portable terminal unit is an embedded system equipped with an RFID reader, GPS, and analog compass as input devices to obtain location and orientation. Finally, it can guide the user to a predefined destination. Custom and dedicated devices are also explored. For instance, in [6], the authors describe a binaural sonar that detects objects over a wider bearing interval and also determines if the object lies to the left or right of the sonar axis in a robust manner. As the sonar or object moves, vibration patterns provide landmark, motion and texture cues. The assistive devices in the shape of a cane has been also extensively tested in [4], [12], and [9]. These devices pack different sensors and feedback to help the blind people to safely navigate in any environment.

Finally, regarding the assistant robots, we can find several examples in the state of the art. For instance, in [7] it is described a proof-of-concept prototype of a robotic shopping assistant for the visually impaired. The purpose of RoboCart is to help visually impaired customers navigate a typical grocery store and carry purchased items. To do so, the robot relies in RFID tags and range sensors. Another examples is [3], that proposes the design of CaBot (Carry-on roBot), an autonomous suitcase-shaped navigation robot that is able to guide blind users to a destination while avoiding obstacles on their path.

3 Personal Assistance Walking System for the Visually Impaired

As explained before, we propose a robotic quadruped that can mimic the abilities of an actual guide dog for the visually challenged. Specifically, we employed a commercial quadruped as the base, and we developed different functionalities so that it can safely walk a person between two points in the space. In this Section, we delve in the hardware modifications and software implementations that we especially tailored for this project.

3.1 Hardware Architecture

The Unitree Go 1 EDU is the robot of choice to carry out this project. This is a mid-size low-cost quadruped that is packed with several sensors and is ROScompliant. It has some delicate parts already working off-the-shelf, as the locomotive and stabilization system. It has fish-eye lens stereo cameras around its perimeter, but the quality of the tridimensional data provided is not accurate. It also has a number of embedded NVidia Jetson Nano and Raspberry Pi devices that control parts of their built-in functionalities. Some of these devices have external USB ports that we used to expand its capabilities. The diagram of the hardware part can be seen in the Figure 1.

In order to achieve the proposed goals, some additions were made to the base of the robot. First, the robot is able to localize itself in an unknown outdoor environment thanks to an additional GPS. The selected model was an U-Blox7 USB GPS module due to its small size and high reliability. It is connected to one of the NVidia Jetson Nano inside the robot.

We also mounted a digital compass to be able to accurately estimate the heading of the robot. In this case, we chose the HMC5883L which is a three-axis magnetoresistive sensor that provides 1° to 2° degree compass heading accuracy. It features an I2C interface so it was connected to an Arduino UNO. The Arduino



Fig. 1. Diagram of the hardware used for PAWS.

queries the compass and sends the data to one of the built-in NVidia Jetson Nano over USB connection.

As the tridimensional data provided by the built-in stereo setup is insufficient to provide accurate and reliable depth that allowed autonomous navigation, we also fixed an OAK-D Pro camera in the neck of the robot. This is a highresolution stereo active sensor that works in the 0.5-12m range. The camera is also able to provide high-resolution color images, which are used for detecting objects as well. It is connected to one of the built-in NVidia Jetson Nano over USB connection.

The last piece of hardware we added is a simple momentary switch. The switch is mounted on a handle that is attached to a robust coiled lead connected to the Arduino UNO mentioned earlier. The intention behind this is two-fold: on one hand, it serves as a security measure, as the robot will only move if the user is pressing the button; and on the other hand, the lead is used so the user can easily follow the steps of the robot.

As metioned before, several devices are connected to a NVidia Jetson Nano using USB. In order to do this, we employed a USB 3.1 hub to expand the number of available ports. We also designed custom 3D-printed parts to hold all the added hardware in place. The plastic parts were fixed on the mounting rails the robot has in its back. Figure 2 shows the ensemble on the actual robot.

3.2 Software Framework

The software we implemented allows PAWS to find a route between two GPS points using the global planning. The actual position is retrieved using the GPS, and the destination is inserted by the user, that is, the place they want to go to. This returns a list of GPS waypoints. The obstacle avoidance and local planning part is in charge of navigating between two subsequent waypoints without colliding against any obstacle. These two main pieces are thoroughly described next in this Section.

As mentioned before, the Unitree Go 1 is ROS-compliant, and thus, all their built-in parts are compatible with ROS. Specifically, we are only interested in the locomotion subsystem that allowed to move the robot by sending velocity



Fig. 2. Close-up picture of the hardware used for PAWS.

commands to the corresponding topic. No low-level interaction is needed to move the legs. Thus, we adopted the methodology and expanded it with new ROS nodes that controlled the newly added hardware parts. Specifically, we created a node that publishes the image and the depth map of the OAK-D Pro in two different topics. We also wrote a node that publishes the GPS coordinates (latitude and longitude). Finally, we also integrated another node that publishes the data retrieved from the Arduino, that is, whether the button of the handle is being pressed, and the heading of the robot (regarding the north) in angular degrees, in another two topics.

All these ROS topics are queried by the higher level parts of the architecture to run their algorithms.

Global planning Regarding this part, we relied in the public Google Cloud Platform API. Once the user inserts the desired destination in GPS coordinates and the robot retrieves the current GPS position, it queries the API for the best route. The computed route is composed of a list of waypoints, as shown in Figure 3, which are also GPS coordinates. Each waypoint represents a turn event, so the path between two of them are basically straight lines. At this point, the navigation algorithm is straightforward: align the robot so it is pointing to the next waypoint and run the obstacle avoidance algorithm until the waypoint is reached. Repeat the process with the next waypoint until the final destination is reached.



Fig. 3. Visualization of the waypoints returned by the Google Cloud Platform API.

Obstacle avoidance As for this method, its goal is to safely move and guide the user between two of the waypoints computed by the global planning method. As explained before, the path between two waypoints is basically straight, but in the real world there are different events that require local navigation and planning that considers the changing state of the environment. As PAWS is aimed to outdoors and urban environments, it can encounter other pedestrians, vehicles, static obstacle like posts or walls, or simply areas that are unsuitable for pedestrians like grass patches. The algorithm should be able to adapt to this challenging setup while navigating to the destination waypoint.

To do so, the algorithm we designed uses as input the GPS position, the heading data and the depth map. GPS position and heading ensure that PAWS is effectively approaching the destination whilst the depth map allows obstacle avoidance.

The algorithm is as follows. At the start, the robot must head the destination waypoint. First, the depth map is retrieved and split in four different vertical parts: the center part, left and right bands, and sides. Figure 4 shows a visualization of this splitting method. The sides of the depth map are discarded, as we are only interested in the area that is right in front of the robot. For each of the parts, we computed the closer depth values, which will represent the nearest obstacle. If there are no obstacles in a range of 3 meters in the center part of the scene, the robot will walk forward at 0.8m/s. If there is an obstacle in the range of 1.5-3m, the robot decreases its velocity to 0.3m/s and starts to turn to the clearer side by observing the left and right areas of the depth map. If it reaches



Fig. 4. Vertical zones defined over the depthmap.

an obstacle under 1.5m, it simply stops for 3 seconds, in the case is a moving object like another pedestrian, to pass. If the obstacle does not move, the robot starts turning around at 0.1m/s looking for a new path. Every 30 frames of going forward, the robot checks its heading and corrects it to point to the destination waypoint if it does not imply having an obstacle under the 1.5m range. In any case, if the robot detects an obstacle under 1.5m in the left of right sides, it turns to the contrary direction to move away of that obstacle. It also checks the current GPS position to detect if the destination waypoint has been reached.

This reactive approach is simple but effective as it is able to quickly adapt its course to the current particularities of changing environments and considers the heading so that the chosen direction ultimately leads to become close to the next waypoint.

Object detection In the list of waypoints returned by the global planning, there could likely be the need of crossing roads. To do this, the robot must approach the crosswalk and wait for the traffic light to turn green before proceeding. Thus, the pedestrian traffic lights and the crosswalks must be detected.

To do so, we rely in the YOLOv8 method [5]. This deep learning-based architecture is based on convolutional neural networks and returns a list of bounding boxes of the detected objects over the input image. It is efficient, configurable and extensible, so we adopted it. Specifically, we used the 'nano' version with an input size of 640×640 pixels as it provides good accuracy while keeping the computation cost at bay.

We trained the architecture to detect three types of objects: crosswalk, green pedestrian traffic light and red pedestrian traffic light. The dataset of choice was

UrOAC [2]. This dataset depicts urban objects captured in a range of different lighting conditions, from bright daylight to low and poor night-time lighting conditions. In the latter, the objects are only lit by low ambient light, street lamps and headlights of passing-by vehicles. The images are taken from the point of view of a pedestrian, which is similar to the point of view of the robot. There are a range of datasets that depict these urban objects of interest, but they are usually captured from the perspective of a car.

Thus, the robot will wait until the pedestrian traffic light turns green as it finds a crosswalk during its navigation. Once it is green, the robot will resume its usual navigation procedures. An example of the outcome provided by the object detection module is shown in Figure 5.



Fig. 5. actual example of the detections provided by the object detection module.

4 Experimentation

In this Section, we put to test different isolated parts of PAWS, and then, we perform an experiment of the whole system in which the robot autonomously navigates to a destination point.

4.1 Obstacle avoidance

To test this module, we execute the obstacle avoidance on the robot in controlled environments. Specifically, three setups were considered with different difficulty level. The first environment (corridor) consisted in a regular building corridor of 1.35m wide per 14 meters long. The second one (maze), was a wider corridor with different obstacles distributed over the space. This case pose a severe challenge for the algorithm due to the density of obstacles and the distance between them. Finally, we run the algorithm outdoors (campus) in our university campus. This is the most similar to a real environment as it has all the urban features. For each environment, we run the algorithm three times, modifying the initial position of the robot to correctly evaluate its behaviour. The number of collisions and the time the robot took to complete the course are reported. These environments are shown in Figure 6.



Fig. 6. Depiction of the different environments in which we tested PAWS.

The results of this experimentation are shown in Table 1. In the easiest setup, that is the corridor, the robot performed as expected. Regardless the starting position it was able to detect the walls and reposition itself to not to collide. It took an average of 45.3 seconds to complete the 14m of corridor.

Regarding the maze, which is the most challenging of the three environments, the time greatly varies depending of the starting point, as the robot takes different decisions reacting to the current state of the obstacles. In any case, the robot took less than a minute to traverse the area. The space between obstacles triggers at all times the reduced approaching speed of the robot as there are obstacles very close. Only one collision occurred and it was because the robot rotated itself on the place and hit the vending machine with the back leg. This happened because the robot was in a very narrow space and due to the lack of peripheral vision as the depth camera faces forward the robot.

Environment	Starting Pose	# Collisions	Runtime
	Center	0	42
corridor	Left	0	51
	Right	0	43
	Center	0	46
maze	Left	0	28
	Right	1	59
	Center	0	60
campus	Left	0	60
	Right	0	60

 Table 1. Results of the experiments carried out to validate the obstacle avoidance module.

The last environment was outdoors, on our University campus. As mentioned, this environment and an actual urban setup are very similar as it features different ground conditions, grass patches, cars, crosswalks, other pedestrians and the rest of features present in an urban setup. In this case, the robot was able to avoid all the obstacles, static and dynamic, present during the test. This environment did not feature narrow places like the ones present in the maze test. The areas were wide and the other pedestrians tend to step aside the path of the robot even before PAWS noticed them. In every case, we limited the time to one minute.

In the light of the experiments, it can be stated that the obstacle avoidance method is good enough for this application, achieving high accuracy in a range of different static and dynamic situations.

4.2 Object detection

The object detection module is in charge of detecting crosswalks, and red and green pedestrian traffic lights. As mentioned before we trained a YOLOv8-nano with the UrOAC dataset for ~500 epochs. The training was stopped when there were no improvements in the validation loss and mean AP metrics for 100 epochs.

Finally, the model performed with good accuracy, reaching a ~0.95 mean AP. Some actual predictions on validation images are shown in Figure 7. Despite the good accuracy, there are some extreme cases in which the detector is prone to fail. For instance, if the crosswalk is partially occluded by bollards, the detector returns different bounding boxes for the same crosswalk as the bollards split them as if they were separate objects. In addition to that, the traffic lights that are far away from the camera are sometimes not detected.

Upon the deployment of the model using PAWS's camera, we realize an issue. The shutter of the color camera is rather slow so the jerky movement of the quadruped introduces artifacts in the image, as the image is being took at the same time the robot is moving. Despite this problem, the model was robust to it, and only affected the smallest objects. If the robot is still, the problem dissapears and the results are as expected.

4.3 Integration of the whole system

In this experiment we tested the system as a whole. The task was to guide a person from a starting point to a destination point. Three different origin points were selected so that they are at a range of around 100 meters. The goal in each case was to reach the main door of our School building.

In all the cases, the robot was able to reach the goal point avoiding collisions with other pedestrians, walls or trees. The time it took to complete the three runs were 47, 59 and 462 seconds. In the second run, PAWS chose to cross a grass field with threes. Despite it is allowed to walk over there in our University, it could be forbidden in other environments. In addition to that, in the same run, the robot tripped on a set of big roots that protrude from the ground. Nonetheless, it did not fall and reached the final destination. However, this could be dangerous



Fig. 7. Results of the object detection module on validation images.

for the person using PAWS at that moment. In the third run, the person that was being using PAWS had to stop for awhile as a colleague interrupted the experiment to chat a little. The user leveraged the control leash to stop the robot and to resume its way later.

5 Conclusion and Future Work

In this work, we propose PAWS, a personal assistance walking system for the visual impaired. Our approaches uses the Unitree Go1 as the base, which a commercial low-cost quadruped. The system leverages a range of added sensors, such as GPS, digital compass and a depth camera, to autonomously navigate between two points in space. To do so, different software components, that interact to each other, were tailored. As the experiments demonstrated, the robot was able to successfully traverse a range of different environments without colliding to any obstacle.

Nonetheless, the approach has some limitations. For instance, the robot lacks of peripheral vision, so there are cases in which it can hit obstacles located behind its body. We also noted a minor problem with the color camera of choice. As the shutter is slow, there appear artifacts in the images if the robot is moving while it is capturing an image. We also plan to implement a semantic segmentation method that classifies what kind of ground is traversable (sidewalk, rugged terrain, and others) or not (grass patches, potholes and others).

To address these problems, we propose the following improvements for future work. First, we plan to append a camera stabilization method that would help to remove the artifacts in the image when the robot is moving fast. In addition, we also plan to append a 360 degrees depth perception device that allows the robot to perceive all its surroundings.

Acknowledgments. This work has been carried out under the framework of the project "SAPEE: Sistema de Ayuda para Personas de la 3^a Edad" which was granted by the International Center for Aging Research in 2023, and with funds of the grant PID2022-1384530B-I00 funded by MICIU/AEI/10.13039/501100011033 and by "ERDF A way of making Europe".

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

- Bauer, Z., Dominguez, A., Cruz, E., Gomez-Donoso, F., Orts-Escolano, S., Cazorla, M.: Enhancing perception for the visually impaired with deep learning techniques and low-cost wearable sensors. Pattern Recognition Letters 137, 27-36 (2020). https://doi.org/https://doi.org/10.1016/j.patrec.2019.03.008, https://www.sciencedirect.com/science/article/pii/S0167865519300881, learning and Recognition for Assistive Computer Vision
- Gomez-Donoso, F., Moreno-Martinez, M., Cazorla, M.: Uroac: Urban objects in any-light conditions. Data in Brief 42, 108172 (2022). https://doi.org/https://doi.org/10.1016/j.dib.2022.108172, https://www.sciencedirect.com/science/article/pii/S2352340922003766
- Guerreiro, J.a., Sato, D., Asakawa, S., Dong, H., Kitani, K.M., Asakawa, C.: Cabot: Designing and evaluating an autonomous navigation robot for blind people. In: Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility. p. 68–82. ASSETS '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3308561.3353771, https:// doi.org/10.1145/3308561.3353771
- Hoyle, B., Waters, D.: Mobility AT: The Batcane (UltraCane), pp. 209–229. Springer London, London (2008). https://doi.org/10.1007/978-1-84628-867-8_6, https://doi.org/10.1007/978-1-84628-867-8_6
- Jocher, G., Chaurasia, A., Qiu, J.: Ultralytics YOLO (Jan 2023), https://github. com/ultralytics/ultralytics
- Kuc, R.: Binaural sonar electronic travel aid provides vibrotactile cues for landmark, reflector motion and surface texture classification. IEEE Transactions on Biomedical Engineering 49, 1173-1180 (2002), https://api.semanticscholar. org/CorpusID:20802580
- Kulyukin, V., Gharpure, C., Nicholson, J.: Robocart: toward robot-assisted navigation of grocery stores by the visually impaired. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2845–2850 (2005). https://doi.org/10.1109/IROS.2005.1545107
- Organization, W.H.: World sight day, vision 2020. https://www.who.int/news/ item/09-10-2003-up-to-45-million-blind-people-globally---and-growing, [Accessed 04-06-2024]

- Penrod, W., Corbett, M.D., Blasch, B.: Practice report: A master trainer class for professionals in teaching the ultracane electronic travel device. Journal of Visual Impairment & Blindness 99(11), 711-715 (2005). https://doi.org/10.1177/0145482X0509901110, https://doi.org/10.1177/ 0145482X0509901110
- Ross, D.A., Lightman, A.: Talking braille: a wireless ubiquitous computing network for orientation and wayfinding. In: Proceedings of the 7th International ACM SIGACCESS Conference on Computers and Accessibility. p. 98-105. Assets '05, Association for Computing Machinery, New York, NY, USA (2005). https://doi.org/10.1145/1090785.1090805, https://doi.org/ 10.1145/1090785.1090805
- 11. Villamizar, L.H., Gualdron, M., Gonzalez, F., Aceros, J., Rizzo-Sierra, C.V.: A necklace sonar with adjustable scope range for assisting the visually impaired. In: 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). pp. 1450–1453 (2013). https://doi.org/10.1109/EMBC.2013.6609784
- Wahab, M.H.A., Talib, A.A., Kadir, H.A., Johari, A., Noraziah, A., Sidek, R.M., Mutalib, A.A.: Smart cane: Assistive cane for visually-impaired people. CoRR abs/1110.5156 (2011), http://arxiv.org/abs/1110.5156
- Yelamarthi, K., Haas, D., Nielsen, D., Mothersell, S.: Rfid and gps integrated navigation system for the visually impaired. In: 2010 53rd IEEE International Midwest Symposium on Circuits and Systems. pp. 1149–1152 (2010). https://doi.org/10.1109/MWSCAS.2010.5548863

Re-engineering EBO: Advancing Social Robotics for Enhanced Care

Sergio Eslava, Alejandro Torrejón, and Pedro Nuñez

RoboLab, Robotics and Artificial Lab Universidad de Extremadura, Escuela Politécnica, Cáceres, España robolab.unex.es

Abstract. This paper outlines the redesign process of the social robot EBOv2, an evolution of the EBO robotics platform. The original version of EBO has been used in various applications, from promoting computational thinking in educational settings to providing cognitive therapy for older adults. The redesign of EBOv2 aims to meet today's needs by incorporating state-of-the-art hardware and a new casing. EBOv2 features differential navigation hardware, an upgraded screen for expressive emotional display, microphones and speakers for personalized user interaction, and optimized components to enhance overall performance. The main goal is to improve the user experience and the robot's versatility for widespread use in occupational therapy, assistance for older adults, and managing neurodegenerative diseases. This article details the co-creation process of EBOv2, focusing on key hardware design aspects, component selection and integration, aesthetic casing design, and comprehensive system integration. Additionally, a group of occupational therapists evaluated the new physical redesign of the robot. This evaluation involved surveys designed by experts in the field to assess the robot's functionality, user acceptance, and overall impact on future therapeutic outcomes. The results highlight significant improvements in the acceptance of the new robot platform, underscoring its potential for broad application in therapeutic and assistive contexts.

Keywords: Social Robotics, Robotics Redesign, Educational Robotic, Prototype generation, Elderly assistance

1 Introduction

As the global population of older adults grows, there is an increasing demand for treatments tailored specifically to their needs. Social robots have emerged as a promising solution, designed to interact with humans in social and therapeutic contexts, providing companionship, entertainment, and support. These robots have effectively delivered therapeutic benefits to older adults [3]. However, traditional approaches often face limitations in personalization, interaction, and engagement. To address these challenges, social robots must have advanced capabilities to facilitate personalized interactions. This includes enabling the robot to engage in natural conversations with each user, understand and respond to their needs and preferences, and provide affective responses. Such capabilities can significantly improve therapeutic outcomes and the overall user experience. This paper presents the hardware redesign process of the social robot EBOv2 to integrate these new capabilities. EBOv2 is an evolution of the EBO robotics platform, which has been utilized in various applications, ranging from promoting computational skills in educational settings to providing cognitive therapy for older adults.

Originally, EBO was designed for limited interaction capabilities; nevertheless, it has been positively utilized in therapies with older adults [13]. EBO used the CORTEX architecture in its latest version to facilitate personalized interactions. It utilized a series of software agents that maintained an engaging and tailored conversational flow using Large Language Models (LLMs). However, the interaction was constrained by the hardware of the original platform, which was designed a decade ago.

The redesign of EBOv2 aims to meet current needs by incorporating stateof-the-art hardware and a new casing. To address the evolving requirements of the EBO robot, several specific needs were identified in a co-creation process: i) the integration of sensors necessary for occupational therapy activities; ii) a larger screen to facilitate use by older adults; iii) a modular design that separates the base from the body to improve access to components; and iv) a uniform finish to enhance aesthetic appeal. EBOv2 features differential navigation hardware, an upgraded screen for expressive emotional display, advanced microphones and speakers for personalized user interaction, and new components to enhance overall performance. These requirements formed the foundation of the redesign process, guiding the selection of features and design elements to enhance the robot's functionality and usability. The main goal of this redesign is to improve the user experience and the robot's versatility for widespread use in personalized cognitive therapies for older adults.

This article details the co-creation process of EBOv2, emphasizing critical aspects of hardware design, component selection and integration, casing design, and overall system integration. Furthermore, a group of occupational therapists has evaluated the new physical redesign of the robot using surveys developed by experts to assess the new embodiment of EBOv2 and its impact on users. The results demonstrate substantial interest in the new redesign, significant improvements in user acceptance, and the anticipated ease of use of the updated robotic platform in therapeutic and care settings.

This article is organized as follows: After this brief introduction, Section 2 reviews related works, focusing on the use of social robots in cognitive therapy for older adults and the challenges they face, including the Uncanny Valley phenomenon and the importance of robot design for user engagement. Section 3 provides a detailed overview of the original EBO robot, including its hardware and software components and its application in therapeutic contexts. Section

presents the redesign of EBOv2, discussing the co-creation process with engineers and occupational therapists, the new hardware components, and the aesthetic improvements in its casing. Section details the evaluation process of EBOv2 by occupational therapists, highlighting the improvements in user acceptance and usability based on survey results. Finally, Section concludes the paper, summarizing the key findings and discussing future work to enhance further the EBOv2 platform for broader application in therapeutic and assistive contexts.

2 Related Works

The use of social robots in cognitive therapy for older adults has been explored in the last decades, with various studies demonstrating their effectiveness in enhancing cognitive functions, reducing depression, and alleviating loneliness. For instance, the study in [2] evaluated the Paro robot, a seal-shaped therapeutic robot, in an Italian Alzheimer's day center. The findings highlighted significant improvements in patients' perceived quality of life when the Paro robot was integrated with usual care, underscoring the potential of social robots in non-pharmacological interventions for dementia. Similarly, in [9], the authors demonstrate the positive impacts on cognitive function, depression, and loneliness among older adults living alone. The intervention used a quasi-experimental design to show significant improvements, reinforcing the therapeutic benefits of social robots. Additionally, the work in [7] investigates the effectiveness of the socially assistive robot Hyodol, finding significant reductions in depression and improvements in cognitive functions among older adults with mild cognitive impairment. Moreover, our study [13] evaluates the earlier version of the EBO robot, demonstrating its positive impact on the interaction and engagement levels of patients with mild to moderate cognitive impairment, further confirming the potential of social robots in the apeutic contexts.

The Uncanny Valley phenomenon, first described by Mori in [11], refers to the discomfort people feel when confronted with robots that appear almost human but not quite. This phenomenon is interesting in the design of social robots, as it impacts their acceptance and effectiveness. Research indicates that robots with a mascot-like appearance, which are less likely to fall into the Uncanny Valley, generally achieve higher acceptance and evoke more positive emotional responses from users. For instance, the work in [1] found that human-like robots often evoke negative emotional responses, hindering their effectiveness in therapeutic settings. Conversely, robots designed with a more friendly, non-humanoid appearance tend to avoid the eerie feeling associated with the Uncanny Valley and are more readily accepted by users. Our original EBO robot addressed the Uncanny Valley phenomenon by adopting a non-humanoid, friendly design [13]. This approach helps avoid the Uncanny Valley and promotes a sense of comfort and control among users. By prioritizing a mascot-like appearance, EBO enhanced its effectiveness in interacting with humans, particularly in therapeutic contexts. In this paper, the new EBOv2 design ensures that the robot can provide emotional support and companionship without eliciting discomfort, making it more suitable for use in occupational therapy, assistance for older adults, and managing neurodegenerative diseases.

The appearance of social robots plays a critical role in user engagement. In [5], the authors highlight that social robots' embodiment and physical design significantly influence user experience and engagement. Their findings suggest that incorporating expressive gestures and a user-friendly interface enhances the quality of interaction between humans and robots. The authors in [8] demonstrate that the robot's appearance plays a crucial role in therapy protocols, indicating that user engagement and therapeutic outcomes are significantly influenced by the robot's design. In the work presented in [10], the authors affirm that social robots can be categorized into several types: socially evocative, socially situated, sociable, socially intelligent, and socially interactive. The level of development of each one of them directly influences user engagement. Therefore, EBOv2 falls under the socially intelligent robots, designed to exhibit human-like social intelligence through advanced human cognition and social competence models. The redesign of EBOv2 focuses on enhancing user engagement by incorporating a larger screen for expressive emotional display, which is particularly beneficial for older adults. This feature, combined with speakers and microphones to facilitate personalized affective interaction, further improves the robot's usability and acceptance.

Finally, social robots should possess the capabilities to interact with affection and empathy during therapy, including understanding and responding to users' emotional states. In the work [12], the author demonstrates that simple emotional gestures in socially assistive robots can significantly increase user engagement and reduce anxiety in medical settings. Similarly, in [4], the authors explore how robot posture and idle motion affect emotional contagion during human-robot interactions, highlighting the importance of affective design in user perception. The EBOv2 redesign incorporates these advanced interaction capabilities at the hardware level, combined with the CORTEX architecture agents [6], ensuring that this robot can meet the diverse emotional and cognitive needs of older adults in various therapeutic settings.

3 EBO: a social robot for therapy with older adults

This section provides a comprehensive overview of the EBO platform, a socialcare robot designed by the RoboLab research group at the University of Extremadura. It builds upon its predecessor, the LearnBot, which was initially developed by the same group in 2005 and serves as a foundational model in educational robotics. The original LearnBot, depicted in Figure 1a, operates on the Odroid-C1 hardware platform and has various sensors and actuators. These include four ultrasonic sensors, a fixed camera, and a differential base, all integral to its functionality.

Figure 1c presents a schematic diagram alongside various real images of the EBO robot, showcasing its design and visual characteristics. These images illus-

trate how the EBO robot displays different emotions on its screen, highlighting its interactive capabilities.



Fig. 1: LearnBot and original EBO.

emotions.

3.1 Hardware components

The original EBO robot was carefully designed on a differential platform, integrating various devices to facilitate environmental perception and expressive capabilities. These features include emotional expressions, image displays on its screen, and physical attributes such as shape, voice modulation, and facial expressions. To ensure that the robot meets the needs and preferences of its intended users—older adults and healthcare professionals—their feedback was actively solicited during the development process. Collaborative meetings and working sessions were conducted to integrate their valuable insights and preferences.

To optimize user acceptance, the external structure of the EBO social-care robot was designed using a prototyping approach. This iterative process enabled continuous refinement based on user feedback. Consequently, the original EBO robot features a sleek, visually appealing design, as shown in Figure 1, where its plastic shape is highlighted. With a diameter of less than 15 cm and a weight of under one kilogram, the EBO robot is compact and lightweight, enhancing its portability and usability.

The EBO robot incorporated hardware components in its original version, specifically focused on therapeutic applications with older adults.

3.2 Software Components: CORTEX architecture

The EBO robot's control software comprises several integral agents that enhance its functionality and interaction capabilities. These software elements manage the physical devices, generate emotional expressions, facilitate human-robot interaction, and display visual information on the robot's screen. In its original version, EBO uses the CORTEX cognitive architecture [6].

- The Navigation agent regulates the robot's forward and rotational movements. It processes commands corresponding to different emotions, allowing the robot to display unique movement patterns that reflect each emotional state. This feature enhances the robot's expressiveness and enables it to adapt its behavior during therapy sessions.
- The Laser agent interprets data from the laser sensors to detect potential obstacles and ensure safe navigation. The robot can move safely and reliably by continuously monitoring its environment, fostering a secure interaction setting.
- The RGB Camera agent manages the servomotor of the camera and employs facial detection algorithms to track users' faces and capture their expressions during therapy. This capability allows the robot to respond effectively to users' emotional cues, promoting a more engaging and personalized experience.
- The Facial Expression agent is vital for generating the robot's emotional expressions. Utilizing various algorithms and models, this component enables the EBO robot to display basic emotions, enhancing its ability to convey empathy and form emotional connections with users.
- The HRI (Human-Robot Interaction) agent enables communication between the EBO robot and its users. It employs Automatic Speech Recognition (ASR) and Text-to-Speech (TTS) algorithms to facilitate speech-based interactions throughout therapy sessions, improving the robot's communication capabilities and ensuring appropriate audio responses.

Upon activation, the EBO robot establishes a local WiFi network that connects it with other interaction components. While designed for autonomous operation, the robot supports teleoperation via an intuitive interface. The EboTalk tool implements a predefined protocol to ensure seamless communication, but it is crucial that the dialogue flow can be modified during supervision. The interface must also support the transmission of emotions and subtle movements, enriching the dialogue with emotional elements.

4 EBOv2: a redesign of a social robot

The redesign of EBOv2 aims to enhance the functionality, usability, and user experience of the original EBO robot. This new iteration addresses today's needs by incorporating advanced hardware components and a user-friendly design. The goal is to create a more effective and versatile robot that can be widely used in occupational therapy, assistance for older adults, and management of neurodegenerative diseases.

The development process of EBOv2 involved two co-creation sessions with both engineers and occupational therapists from the research team as well as various senior centers. These sessions were pivotal in identifying key requirements and ensuring the robot's design met the specific needs of its intended users. The engineers focused on integrating state-of-the-art hardware, while the occupational therapists provided their expertise on the practical applications of the robot and the interaction needs required for it.

EBOv2 was presented in workshops with older adults and occupational therapists to refine the design further¹. These sessions provided feedback on the robot's appearance, functionality, and overall user experience. Participants interacted with various prototypes and provided their opinions on the physical design, including the casings' shape, size, and aesthetic appeal.

These workshops and co-creation stages revealed preferences and suggestions from the end-users, which were then incorporated into the final design. The feedback highlighted several key aspects: the importance of a compact and lightweight form factor, a larger high-resolution screen to display emotions better and facilitate interaction, enhanced microphones and speakers to improve communication and interaction quality, and the need for a user-friendly interface that older adults could easily navigate, preferably through voice commands.

4.1 Hardware components

This section will define four types of elements: controller, actuators, sensors, and power supply, which are described below.

Controller EBOv2 has an Jetson Xavier NX as a controller due to its AI and machine learning capabilities and energy efficiency, essential for real-time image processing, emotion detection, and extended operational time. Its GPIO ports provide flexibility for integrating various sensors and actuators, facilitating seamless communication between components. Additionally, the robust development ecosystem from NVIDIA supports rapid implementation and optimization of AI applications. The scalable architecture of the Jetson Xavier NX allows for future upgrades, ensuring the long-term viability and adaptability of the EBOv2 robot.

Actuators Among the actuators that will enable human-robot interaction are:

- The differential wheels ensure smooth and fluid movement of the robot.
- LED lights, model WS2812B, this type of LED allows for individual RGB interaction, enabling the display of complex and dynamic patterns or interactions to express emotions or actions.

¹ Generación Silver: Avances en Robótica Asistencial e Inteligencia Artificial, Cáceres, Spain, may-2024
7-inch touch screen with speaker, the increased screen size facilitates viewing items or substitutes in case of hearing difficulties, adding the complement of touch interaction.

Sensors The sensors for observing the external word are:

- Lidar VL6180X, with 5-200 millimeters, this sensor helps the robot avoid obstacles.
- RGB sensor TCS34725, located at the bottom of the robot, to carry out exercises in educational robotics, such as line following or color differentiation.
- RGB camera with 130 degrees FOV on the front, to human and object recognition.
- Microphone omnidirectional array XVF-3000 to speech recognition.

Power supply The electrical requirements for EBOv2 include ensuring an effective autonomy of at least 4 hours and powering the hardware elements of the robot, which have an average consumption of 35W and a peak consumption of up to 60W. To meet these demands, the system is equipped with one 5V power bus for the LEDs and one 9V power bus for the motors and the controller, as the controller operates within a range of 9-19V. The controller's power supply is connected to the 9V source, reducing material costs and saving space. Considering these requirements, the following components have been selected to fulfill these needs:

- 144 Watts, 24 volts Li-ion battery. This battery provides the necessary power for the robot's operations.
- 5 Amperes fuse to ensure safety by protecting the electrical components from overcurrent.
- Converter 24 volts to 9 volts and 5 volts to adapt the voltage from the battery to the required levels for different robot components.

This ensures that the electrical demands are met and allows for the implementation of modern systems without concerns about the power supply.

Appendix A shows the electrical schematic developed for this purpose.

4.2 Software Components

The software components of the EBOv2 robot remain largely unchanged, as the existing software developed for the CORTEX architecture is fully compatible with this new version. This continuity ensures that all previously developed functionalities and interaction capabilities are preserved and optimized for the EBOv2 platform.

4.3 Body Design

The design philosophy of EBOv2 focuses on creating a functional and emotionally engaging robot. The robot's appearance is crucial in its acceptance and effectiveness, particularly in therapeutic and educational settings. The body design process incorporates several key elements to achieve this balance: **Prototyping Process** Initially, rapid prototyping was conducted to explore and refine key concepts for the robot's body design. This phase involved creating multiple prototype versions to visualize different design ideas and assess their feasibility. As shown in Fig. 2, the initial prototypes (Fig. 2a) facilitated the iterative process of experimenting with various shapes and forms. Feedback from these iterations gathered through co-creation sessions with engineers, occupational therapists, and older adults was instrumental in refining the design. The final prototype (Fig. 2b) was selected based on its adherence to the original design's friendly and approachable appearance, aesthetic suitability, ease of use, and overall user acceptance. This collaborative and iterative design process ensured that the EBOv2 met technical specifications and addressed its intended users' practical needs and preferences.



Fig. 2: Prototypical versions of the new EBO design

Figure 3 shows two versions of the new EBO robot from different views. These two versions have been evaluated in the Results section.



Fig. 3: Possible final versions of EBOv2 design

5 Evualuating the EBOv2 social robot

The final design was chosen based on its ability to meet all necessary design requirements while closely adhering to the structure of the original EBO robot. This version ensures that the new EBO remains recognizable and retains the qualities that made the original design successful.

In this phase of the design process, we sought to validate the new design by surveying occupational therapists. Forty occupational therapists working with older adults were selected for the survey. A Likert-type questionnaire encompassed various aspects such as the robot's general appearance, shape, color, and usability in their therapies. The results are presented in Figures 4 and 5.



Fig. 4: Survey results validating EBOv2 design



Fig. 5: Points for improvement highlighted by therapists

The survey analysis indicated strong therapist validation, as shown in Figure 4. Therapists particularly appreciated the robot's familiar and approachable design, essential for user acceptance. Additionally, specific points for improvement were highlighted, as shown in Figure 5, providing valuable insights for further refinement of the robot. This feedback underscores the importance of user-centered design and continuous iteration to ensure the robot meets its users' practical needs and preferences.

Therapists' evaluations of specific aspects of the EBOv2 design showed that while the general appearance, shape, and sense of security were positively received, the chosen color received neutral feedback.

5.1 Comparison

The evolution of the EBO robot, as illustrated in Table 1, highlights significant advancements in technology and design from the original LearnBot to the new EBOv2.

Aspect	LearnBot	EBO Original	EBOv2	
Controller	Odroid-C1	Raspberry PI 3	Jetson Xavier NX	
Sensors	4 ultrasonic sensors	4 single-point LiDAR and Microphone	10 single-point LiDAR and Microphone Array	
Movement	Movement Differential base		Differential base	
Vision	Fixed camera	Fixed camera Mobile camera Fixed wide-angle		
HRI	Movement	Movement, sound and 3" screen	Movement, sound, 7" screen and LED lights	
Programming Interface Robocomp		Robocomp and LearnBlock	Robocomp and LearnBlock	
Simulation Support	No	RCIS	Webots with Robocomp framework	
Materials and Manufacture	3D printed using PLA	3D printed using PLA	3D printed using Recicled PLA	

Table 1: Comparison between LearnBot, EBO Original, and New EBO

One of the most notable upgrades is in the controller. The transition from the Odroid-C1 in LearnBot to the Raspberry PI 3 in the original EBO and finally to the Jetson Xavier NX in EBOv2 reflects a substantial increase in computational power and capabilities. This progression allows EBOv2 to perform more complex tasks, including advanced AI and machine-learning applications for human-robot interaction during therapies.

In terms of sensory capabilities, EBOv2 demonstrates a marked improvement. The original LearnBot's reliance on four ultrasonic sensors has been replaced by a more sophisticated array of sensors in the later models. The original EBO incorporated four single-point LiDAR sensors and a microphone, while EBOv2 features an enhanced setup with ten single-point LiDAR sensors and a microphone array. This significant upgrade enhances the robot's environmental perception and interaction quality, making it more adept at navigating complex environments and understanding auditory cues.

Furthermore, the human-robot interaction capabilities have been expanded. While the LearnBot was limited to basic movement, the original EBO included movement, sound, and a 3" screen; the EBOv2 included a 7" screen, and LED lights on the bottom of the robot. These additions significantly enhance the robot's ability to convey emotions and information, providing a richer, more engaging user experience. Additionally, EBOv2's vision system now includes a fixed wide-angle camera, improving its ability to capture and process visual information compared to the mobile camera of the original EBO.

The programming interface and simulation support have also seen advancements. The consistent use of Robocomp and LearnBlock provides a robust platform for developing and testing new applications. However, transitioning from RCIS to Webots with the Robocomp framework for simulation support in EBOv2 offers more advanced and realistic simulation capabilities, facilitating better preparation and testing of the robot's functions before deployment.

Finally, EBOv2's commitment to sustainability is reflected in its use of recycled PLA for 3D printing, compared to the PLA used in earlier models. This reduces environmental impact and aligns with growing trends in eco-friendly manufacturing practices.

6 Conclusions

In this work, we have explored the iterative design process of the EBO robot, highlighting its evolution from the LearnBot to the original EBO and finally to the new EBOv2. Our primary objective was to maintain the core principles and design features that contributed to the success of the original EBO while incorporating necessary advancements to meet current user needs and technological standards.

The process began with rapid prototyping to visualize and assess various design ideas. This phase allowed us to quickly iterate and refine the robot's body design, ensuring it remained friendly and approachable. By engaging occupational therapists in the evaluation process, we obtained valuable feedback on the design's applicability and effectiveness within its intended deployment context. The feedback validated the new design while pointing out improvement areas and guiding our final adjustments.

Through the comprehensive comparison of LearnBot, the original EBO, and the new EBOv2, we demonstrated significant improvements in hardware, programming interface, simulation support, and HRI. EBOv2 not only builds upon the strengths of its predecessors but also introduces enhanced accessibility and aesthetic features, which are crucial for its role as a socially intelligent robot. In this way, EBOv2 represents a balanced blend of innovation and tradition. It respects the validated design of its predecessor while addressing modern requirements and user expectations. This thoughtful evolution ensures that EBO continues to be an effective and engaging tool in educational and therapeutic settings, fostering positive human-robot interactions and enhancing the overall user experience.

The successful integration of feedback and iterative design has resulted in a functional, practical, appealing, user-friendly robot. As we continue to develop and refine social robots like EBO, the insights gained from this project will inform future innovations and contribute to the growing field of human-robot interaction.

7 Acknowledgements

This work has been partially funded by the PDC2022-133597-C41 project and TED2021-131739-C22 R+D+i projects, supported by Spanish MICIU/AEI/10.13039/501100011033 and the European Union's "NextGenerationEU"/PRTR, by the PID2022-137344OB-C31 R+D+i project, supported by Spanish MICIU/AEI/10.13039/501100011033 "FEDER Una manera de hacer Europa", "FEDER/UE", and by the FEDER Project 0124_EUROAGE_MAS_4_E (Interreg VI-A Portugal-España (POCTEP) Program 2021-2027).

References

- C. Bartneck et al. "Is the Uncanny Valley an Uncanny Cliff?" In: Proceedings of the 16th IEEE International Symposium on Robot and Human Interactive Communication (2007), pp. 368–373. DOI: 10.1109/ROMAN.2007. 4415111. URL: https://ieeexplore.ieee.org/document/4415111.
- [2] Roberta Bevilacqua et al. "Social robotics to support older people with dementia: a study protocol with Paro seal robot in an Italian Alzheimer's day center". In: Frontiers in Public Health (2023). DOI: 10.3389/fpubh. 2023.1141460.
- [3] Cynthia Breazeal. Designing sociable robots. MIT press, 2004.
- [4] Isabel Casso et al. "The Effect of Robot Posture and Idle Motion on Spontaneous Emotional Contagion during Robot-Human Interactions". In: arXiv.org abs/2209.00983 (2022), pp. 1–8. DOI: 10.48550/arXiv.2209.00983.
- [5] Ana Corrales-Paredes, María-José Terrón-López, and Verónica Egido-García. "User Experience Design for Social Robots: A Case Study in Integrating Embodiment". In: Sensors 23 (2023), p. 5274. DOI: 10.3390/s23115274.
- [6] Juan Carlos García García et al. "Towards the design of efficient and versatile cognitive robotic architecture based on distributed, low-latency working memory". In: Proceedings of IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC 2022). Santa Maria da Faira, Portugal, 2022.
- [7] Su Kyoung Kim et al. "Investigating the effectiveness of Socially Assistive Robot on Depression Cognitive Functions of Community Dwelling Older Adults with Cognitive Impairments". In: Assistive Technology (2023). DOI: 10.1080/10400435.2023.2237554.
- [8] Breanna Lee, Jin Xu, and Ayanna M. Howard. "Does Appearance Matter? Validating Engagement in Therapy Protocols with Socially Interactive Humanoid Robots". In: *IEEE Symposium Series on Computational Intelligence*. 2017, pp. 530–537. DOI: 10.1109/SSCI.2017.8285303.
- Jun Seo Lim. "Effects of a Cognitive-Based Intervention Program Using Social Robot PIO on Cognitive Function, Depression, Loneliness, and Quality of Life of Older Adults Living Alone". In: *Frontiers in Public Health* (2023). DOI: 10.3389/fpubh.2023.1141460.

- E. Mahdi et al. "Categorizing Social Robots: A Comprehensive Review". In: Journal of Human-Robot Interaction 15 (2022), pp. 78–93. DOI: 10.1016/j.jhri.2022.03.001.
- [11] M. Mori. "The Uncanny Valley". In: Energy 7 (1970), pp. 33–35.
- [12] Anouk Neerincx et al. "The Effect of Simple Emotional Gesturing in a Socially Assistive Robot on Child's Engagement at a Group Vaccination Day". In: Proceedings of the IEEE/ACM International Conference on Human-Robot Interaction. 2023, pp. 102–113. DOI: 10.1145/3568162. 3576960.
- [13] M.T. Rodríguez-Domínguez, M.I. Bazago-Dómine, M. Jiménez-Palomares, et al. "Interaction Assessment of a Social-Care Robot in Day center Patients with Mild to Moderate Cognitive Impairment: A Pilot Study". In: *International Journal of Social Robotics* 16 (2024), pp. 513–528. DOI: 10. 1007/s12369-024-01106-4. URL: https://doi.org/10.1007/s12369-024-01106-4.

A EBOv2: Electrical Schematic



31

Enhancing Cognitive Therapies for Older Adults with EBO: An Autonomous Social Robot System Integrating LLMs and Therapist-in-the-Loop

Antonio Blanco-Castillo, Jackie Lee Chong Ojeda, Alicia Condón, and Pedro Núñez^[0000-0002-3615-8833]

> Universidad de Extremadura, Spain pnuntru@unex.es http://robolab.unex.es

Abstract. The growing population of older adults necessitates effective cognitive therapies. Social robots have potential as therapeutic tools, providing companionship, entertainment, and support. Traditional methods for cognitive engagement need more in terms of personalization and interaction. This paper presents the development of a system using the social robot EBO for interactive cognitive games and therapies, leveraging Large Language Models (LLMs). The system integrates the narrative generation via Llama on the CORTEX architecture, user response capture with Whisper, and narration through the MeloTTS engine. Tailored for older adults with mild to moderate cognitive impairment, it personalizes interactions based on individual interests, preferences, and cognitive levels. The therapist-in-the-loop approach allows the therapist to participate in the initial game configuration. The system conducts the therapy sessions autonomously, providing a final summary for analysis. This personalized therapy aims to enhance user experience and maximize the therapeutic utility of the system in promoting cognitive and emotional well-being within this demographic group. We evaluated various opensource LLMs and demonstrated their potential effectiveness through a pilot study, showcasing enhanced user experience. This approach could significantly enhance therapies for older adults using social robots as interactive tools.

Keywords: Social robots · Cognitive therapies · Older adults care.

1 Introduction

As the global population ages, the prevalence of cognitive impairments among older adults continues to rise, posing significant challenges for healthcare systems worldwide. Cognitive impairments, which range from mild cognitive decline to severe dementia, impact millions of elderly adults, reducing their quality of life and increasing the burden on caregivers and healthcare providers. Addressing these challenges requires innovative approaches that leverage technological advancements to provide effective, personalized, and scalable cognitive therapies. Social robots have emerged as promising tools in cognitive therapy for older adults. These robots, designed to interact socially with humans, offer companionship, entertainment, and therapeutic support [10]. They can play a crucial role in cognitive engagement through various activities, including interactive storytelling, memory games, and other cognitive exercises. However, traditional methods of cognitive engagement often fall short in terms of personalization and interaction, which are critical for maintaining the interest and motivation of users with cognitive impairments.

This paper presents the development of an advanced system utilizing the social robot EBO for delivering interactive cognitive games and therapies. EBO, originally designed to promote computational thinking in educational settings, has been significantly upgraded to meet the current needs of cognitive therapy. This includes the integration of the CORTEX architecture [5], a robust cognitive framework that supports the deployment of various AI agents to enhance the robot's interactive capabilities. Our system employs Large Language Models (LLMs) such as Llama [12] for generating engaging narratives and interactions. Additionally, it incorporates Whisper [9] for capturing and transcribing user responses and the MeloTTS engine [14] for synthesizing natural-sounding speech. These technologies enable EBO to deliver personalized and adaptive cognitive therapies tailored to the individual interests, preferences, and cognitive levels of older adults with mild to moderate cognitive impairments.

A key feature of our approach is the "therapist-in-the-loop" methodology. This involves the therapist in the initial configuration of the cognitive games and therapies, ensuring that the activities are aligned with each user's specific needs and capabilities. Once configured, the system operates autonomously, conducting therapy sessions and providing a comprehensive summary of the interactions for further analysis by the therapist. This approach enhances the therapy's personalization and maximizes its therapeutic utility by allowing continuous adaptation and improvement based on user feedback and performance.

The primary aim of this system is to enhance user experience and promote cognitive and emotional well-being within the target demographic group. By leveraging advanced AI technologies and a user-centered design, we aim to provide a solution that is both effective and enjoyable for older adults, helping to mitigate the effects of cognitive decline and improve their overall quality of life.

In this paper, we will detail the development and implementation of the EBObased system, discussing its architecture, functionalities, and the specific AI techniques employed. We will also present the results of evaluating different LLM models and a pilot study conducted with testers and caregivers, demonstrating the system's effectiveness and potential as a therapeutic tool.

2 Related works

The utilization of social robots in cognitive therapy for the elderly has gained significant attention in recent years due to the increasing prevalence of cognitive impairments among older adults. Social robots have been developed to provide mental support, companionship, and cognitive stimulation to elderly individuals, particularly those with dementia. For instance, robots like Paro and other interactive devices have demonstrated effectiveness in enhancing the emotional well-being of elderly patients through social interaction and cognitive engagement. These robots can support various therapeutic activities, such as memory games and storytelling, which are crucial for maintaining cognitive functions. However, challenges remain in customizing these interactions to cater to the individual needs of users, as traditional approaches often lack the necessary personalization and adaptability required for diverse cognitive impairments [11]. Additionally, studies have shown that social robots can reduce stress and improve social interaction among elderly individuals [1]. Novel robots like Rassle have been developed to engage elderly users through touch-based interactions, improving their quality of life [15]. Another study highlights the differential efficacy of social robots based on various neuropsychiatric profiles, demonstrating significant improvements in emotional well-being [3].

Automatic Speech Recognition (ASR) and Text-to-Speech (TTS) technologies significantly impact the effectiveness of interactions between social robots and elderly users. ASR enables robots to understand and respond to spoken language, making interactions more intuitive and user-friendly. On the other hand, TTS allows robots to communicate naturally with users by generating human-like speech. Studies have shown that integrating ASR and TTS technologies into social robots enhances their ability to engage elderly individuals by providing clear, understandable, and interactive communication. This integration helps reducing the cognitive load on users, facilitates smoother interactions, and makes the therapeutic experience more engaging and effective [7]. Moreover, using TTS with emotional recognition capabilities allows robots to adapt their responses based on the user's emotional state, enhancing the overall interaction quality [4]. While there are challenges in ensuring the naturalness and accuracy of ASR and TTS technologies, advancements in these areas continue to improve the effectiveness of social robots in therapeutic settings.

Comprehensive systems that combine ASR, TTS, and Large Language Models (LLMs) are emerging as powerful tools in cognitive therapy for the elderly. These integrated systems leverage advanced AI to provide personalized and adaptive interactions, offering a more tailored therapeutic experience. For example, combining ASR, TTS, and LLMs such as GPT-3 enables robots to generate meaningful and contextually relevant responses, enhancing engagement and cognitive stimulation. Such systems have been shown to maintain user interest over extended periods and provide more effective cognitive engagement than traditional methods. These advancements address limitations in previous robotic systems, which often struggled with maintaining long-term engagement due to repetitive interactions [13]. Additionally, personalized cognitive stimulation programs that leverage AI have been proven to maintain normal cognitive functioning and delay cognitive decline, demonstrating the potential of AI-enhanced therapy [6].



Fig. 1. Overview of the proposal described in this paper.

The EBO-based system presented in this paper represents a significant advancement in integrating these technologies for cognitive therapy. EBO can deliver highly personalized and adaptive cognitive therapies by incorporating the CORTEX cognitive architecture and utilizing LLMs like Llama and advanced ASR and TTS systems. The "therapist-in-the-loop" methodology ensures that the therapy is tailored to the specific needs of each user, enhancing its effectiveness and user satisfaction. This approach addresses traditional methods' limitations and sets a new standard for deploying social robots in cognitive therapy. Integrating these technologies allows EBO to provide a more dynamic and engaging therapeutic experience, potentially improving older adults' cognitive and emotional well-being with mild to moderate cognitive impairments [2]. Furthermore, AI-driven personalization in therapeutic programs, such as physical exercise for elderly patients with cancer, has shown significant benefits, highlighting the importance of tailored interventions [8].

3 Self-adaptive Cognitive Therapies with EBO Robot

The system presented in this paper integrates new human-robot interaction capabilities into the Cortex architecture that autonomously adapts therapy to older adults. This section provides a detailed overview of the system's architecture, functionalities, and AI techniques to enhance user engagement and therapeutic effectiveness. Fig. 1 shows a general schematic of the proposed system. First, the occupational therapist selects the type of game that will be developed in the therapy session. Next, the initial game prompt is generated and played through the TTS using the personalized data for each user. At this point, the iteration between the user and the robot begins and continues until the end of the session. During this process, MeloTTS, Whisper, and Llama capture data and update the robot's knowledge through the Cortex architecture. Once the game is completed, the therapist receives feedback from the session.



Fig. 2. The EBO social robot and some of the facial expressions.

3.1 System Architecture

The proposal is based on the Cortex cognitive architecture [5]. This architecture is based on a shared Working Memory (WM) representing the EBO robot's knowledge about the world. Different software agents access this WM to update or use this data to infer new knowledge. Thus, the main components of our system include the EBO robot hardware, the CORTEX cognitive architecture, and several AI-based agents for speech recognition, natural language processing, and adaptive interaction.

a) EBO robotic platform. EBO is a social robot designed in RoboLab, University of Extremadura. It is equipped with sensors, cameras, and actuators to interact with users dynamically and responsively. Its design incorporates expressive features to enhance social interaction and engagement. This ability to express emotions has proven highly beneficial in cognitive therapy with older adults, making a significant difference when our system is implemented in a robot, leading to more engaging interactions [10]. In this same study, we found that it has been very well received, having been tested with real users who provided positive evaluations and expressed a strong desire to engage with it.

Regarding the hardware, a co-creation methodology was employed to design the external form, resulting in a plastic casing with a diameter of less than 15 cm and a weight of under one kilogram to optimize user acceptance. The EBO robot's components include:

- A Raspberry Pi 3B+ that manages the control system for other hardware elements.
- A speaker and microphone for user interaction.
- A camera to capture visual data.
- A display screen to show images, such as emotional expressions.
- Laser sensors that provide distance measurements from surrounding objects.

b) Cortex architecture: Integrating AI Agents for Enhanced Adaptive Interaction. The CORTEX cognitive architecture offers a structured approach for designing, modularizing, and representing robotic activities and data [5]. Within this architecture, agents are tasked with specific roles and collaborate by utilizing a Working Memory. This WM holds the robot's knowledge, which can be predefined, derived from sensor inputs, or generated through agent actions. The WM is organized as a directed graph, where vertices represent metric or symbolic information and edges denote geometric or logical relationships. In this graph, vertices (or nodes) symbolize ontology concepts, while edges define their connections.

Using the CORTEX architecture, agents can share information and coordinate their actions to facilitate self-adaptive human-robot interaction during therapy sessions. Fig. 3 illustrates the CORTEX instance employed in this study. This working memory holds data from various agents, each contributing to the system's overall functionality and interaction capabilities. The agents include:

- Automatic Speech Recognition (ASR) Agent: This new agent accurately captures and interprets spoken language from users, which enables natural and fluid interactions between the user and our robot. In our proposal, the ASR agent uses Whisper, developed by OpenAI [9]. Whisper is an ASR system trained on 680,000 hours of multilingual and multitasking supervised data collected from the web.
- Text-to-Speech (TTS) Agent: The TTS Module synthesizes natural-sounding speech, enabling the robot to communicate effectively and empathetically with users. This capability is achieved by incorporating the MeloTTS technology [14]. MeloTTS, developed by MyShell.ai, is a text-to-speech library that converts written text into natural, fluent speech across several languages. Its primary feature is the generation of highly natural and expressive voices using deep learning techniques.

The MeloTTS system operates through four main stages: first, the input text is processed to extract linguistic structures. Next, intermediate acoustic features representing the speech are generated. These features are then used to synthesize the final audio waveform. This architecture enables real-time synthesis on CPUs without requiring GPUs, ensuring efficient and highquality speech output.

- Dialogue manager Agent: This agent, powered by Large Language Models (LLMs), generates meaningful and contextually appropriate responses during interactions. This enhances the robot's ability to engage users in therapeutic conversations and activities. In our proposal, we used a Llama developed by Meta [12] as our LLM and tested it with different models to adjust the conversations and games developed.
- Player and Game Data Integration (therapist-in-the-loop): This agent includes data relating to the older adult (player) and the selected game base, all through a user application specifically designed for the occupational therapist. This data ensures that the game interactions are tailored to the specific user, considering their cognitive abilities and personal preferences.



Fig. 3. The CORTEX architecture and the set of agents used in this proposal.

- Prompt Generator Agent: The working memory's data allows for the generation of personalized dialogs through the Prompt Generator Agent. This agent synthesizes the information from all sources (including the environment and the user's state) to create prompts for the LLM Agent that guide the interaction, ensuring it is engaging and suitable for the user's cognitive level.

These agents must be interconnected precisely, where CORTEX plays a crucial role. By reading real-time information from the graph, CORTEX facilitates this interconnection. Each agent will have a node with different attributes that change as the system operates. Other agents will detect changes in these attributes to determine when to act. The Working Memory is described in Fig. 4, where nodes and their associated attributes are detailed.

Interconnection of Agents to Ensure System Functionality The interconnection of the agents is designed to facilitate seamless communication and operation within the AI module system. Table 1 outlines the processes involved in this interconnection, illustrated in Fig. 5.

This structured interconnection ensures continuous and efficient agent interaction, enabling the system to function smoothly throughout the game.

4 Interactive Cognitive Games and Therapies: therapist-in-the-loop

The system features cognitive games designed to enhance memory, attention, and problem-solving skills, through interactive storytelling, task memorization,



Fig. 4. The Working Memory defined in our use case. The CORTEX architecture implements three main nodes for interconnecting the ASR, TTS, and LLM agents.

and puzzles. Tailored to individual cognitive levels and interests, these games offer personalized and engaging experiences. Occupational therapists play a key role by inputting user data, selecting games, adjusting difficulty, and providing feedback. They can also alter the game flow via an interface connected to the Working Memory using a designated agent.

This paper details the implementation of three games focusing on verbal training for Activities of Daily Living (ADLs), which are critical for maintaining and enhancing the independence of older adults. ADLs refer to the essential daily tasks necessary for individuals to live autonomously and fulfill basic needs.

Game 1: ADLs Sequences This activity aims for the participant to correctly sort the steps of a basic ADL that EBO will present in an unsorted sequence. EBO will begin by announcing the activity and then randomly providing the steps. The participant's task is to verbally sort these steps correctly, ensuring they reflect the proper sequence for the activity. EBO will evaluate the answer and provide feedback to determine if the response is correct. Table 2 presents the initial prompt and the commencement of the game.

Game 2: True or False of ADLs The game aims for the participant to identify whether the statements related to basic daily life activities provided by EBO are true or false, reinforcing their knowledge and understanding of these activities. EBO will present a series of statements about various basic daily life activities, and the participant must determine if each statement is true or false. This exercise tests their comprehension and helps them retain critical information about daily living tasks. Each game will consist of five questions, with the correct answers recorded for further analysis and feedback. Table 3 shows this game's initial prompt and the first user's interactions.

Process	Description
Initialization and Response Gener-	
ation	
	 The LLM agent receives the initial prompt from the Game
	agent and stores it in the LLM 'in' attribute.
	 The LLM agent then generates a response to initiate the game, storing this response in the LLM 'out' attribute.
TTS Processing	
	 The TTS agent monitors the LLM 'out' attribute. Upon detecting a change, the TTS 'to say' attribute is updated with the new response. The TTS agent is programmed to add the string from TTS 'to say' to the playlist upon detecting this update. Once the message finishes playing, the TTS agent sets the ASR 'listening' attribute to True.
ASR Processing	
	 The ASR agent begins listening when it detects that the ASR 'listening' attribute is activated. After capturing the spoken input and detecting silence, the ASR agent updates the ASR 'text' attribute with the tran- scribed string.
Loop and Continuation	
	 The LLM agent re-enters the process by detecting the change in the ASR 'text' attribute. The LLM agent updates the LLM 'in' attribute with the newly transcribed text from the ASR agent. The system then reverts to the beginning of the loop, where the LLM agent generates a new response based on the up- dated prompt, continuing this cycle until the game concludes.

Table 1. Overview of the EBO Robot's Interactive System Processes

Game 3: The Shopping The game aims for the participant to identify the item that should not be present from a list of ingredients required to achieve a specific goal. Once correctly identified, they must calculate the total cost of the purchase in euros. This aids in improving planning, money management, and mathematical skills. EBO will start by announcing the meal to be prepared, and the participant will select the unnecessary item. Once all ingredients are chosen, EBO provides their prices in euros. With this information, the participant calculates the total cost of the purchase. Upon completion, EBO assesses the list and the calculation and offers its opinion to the participant. Table 4 shows the initial question and the start of this game

5 Experimental results and discussion

5.1 Comparison of the different LLM mo dels evaluated

In this subsection, we will analyze the different LLM models we have tested, with the help of professional occupational therapists working in the group, to determine which one best meets our needs. Our evaluation will focus on each model's ability to generate engaging and effective games. By comparing their performance, we aim to identify and implement the model that best enhances our system and user experience.

To analyze and compare the different models studied, we focused on asking them to generate a single game: the sequences of daily life activities. Specifically,



Fig. 5. Schematic showing the interconnection of the AI agents



Fig. 6. The figure illustrates the cognitive therapy process with the EBO robot, showing the initial prompt of the game-generating agent along with the development of the game that leverages ASR/TTS or working memory.

Initialization prompt	I want you to play with me. You must present a scrambled sequence of steps to reach the goal that I indicate. I'll respond one by one. You tell me if it's correct or not. If it's correct, we continue. If it's wrong, we start over. I'll provide key data in JSON format. Customize the game based on that data. Don't tell me anything now. Just start the game
LLM response	Welcome to the game, Maria! Here's your first scrambled sequence of steps to prepare orange juice. Add sugar to taste. Cut the oranges in half. Wash the oranges. Squeeze the juice from the oranges. Pour the juice into a glass. What's your first move?"
First user intervention	Add sugar
Game continuation	Sorry, that's not the correct next step. Let's start over. Here's the scrambled sequence again: \ldots

 Table 2. Initial prompt and fist interactions of the game designed for the ADLs sequences

Initialization prompt	I want you to play with me. You have to present me with a question related to a daily life activity that gets me to the goal I present. I'll respond true or false. You tell me if it's correct or not. If correct, we continue. If wrong, we start over. I'll provide you with the most important data in JSON format. Personalize the game according to that data. Don't tell me anything now. Just start the game.
LLM response	María. Let's start the "ADL True or False" game by preparing orange juice in the kitchen with your sister Anal Question 1: First, do you need to select ripe oranges from the fridge? True or false?
First user intervention	True
Game continuation	Step 2: Great! Let's move on. Then, do you wash the oranges under the faucet? True or False?

Table 3. Table presenting the game designed for the True or False game of ADLs

we asked the LLM models to generate a 5-step sequence, and we tested this with three different models. The model parameters have been carefully selected to ensure the most appropriate and accurate outputs possible.

The three models under study are:

Llama 3

It'll be used Meta-Llama-3-8B-Instruct, a fine-tuned version specialized for instructional text generation, that will be executed on one of the two A100 80GB GPUs available in a rack server. The following key parameters are configured:

- temperature=0.6: is applied during text generation to control the randomness and ensure conservative outputs.
- top_p=0.9: Top-p probability is used for sampling, allowing for diverse outputs while maintaining coherence.
- max_gen_len=None: refers to the maximum generation length, meaning there is no explicit limit on the length of generated sequences.

The *tokenizer.model* is used to tokenize the input text into a format that the model can understand. It contains the vocabulary and rules necessary for converting raw text into numerical inputs that the model can process.

Initialization prompt	I want you to play with me. You have to present me with a shopping list to achieve the goal I indicate, with one false item. I have to find the false item. You tell me if it's correct or not. If correct, we continue, and you give me the prices in euros so I can calculate the total. If wrong, we start over. I'll provide you with the most important data in JSON format. Customize the game according to those data. Don't tell me anything now. Just start the game.
LLM response	Welcome, María! We're about to start our shopping game. Your goal is to prepare orange juice. Here's your first shopping list: Oranges Sugar Ice Lettuce Lemons Can you identify the false item ?"
First user intervention	Lemons
Game continuation	Correct! Lemons don't belong in orange juice. Well done! Now, let me provide you with the prices in euros for the items on your list: Oranges - €2.50 Sugar - €1.20 Ice - €0.80 Lettuce - €1.00 Please calculate the total.

Table 4. Table presenting the game designed for The Shopping game

ChatQA-1.5-8B-Q8 0.gguf

It has been downloaded from Huggingface repository and loaded using LlamaCpp. During the conversation, a prompt template defines the structure of interactions, enabling seamless communication between the user and the AI assistant.

The model is triggered with user responses to generate coherent and relevant answers. As part of the setup, a specified embeddings model is initialized, responsible for converting text inputs into numerical representations. Additionally, the conversation history is managed using Chroma DB, which stores text embeddings for efficient retrieval and context maintenance. This testing has been conducted on a local computer. The system is equipped with 32.0 GiB of RAM, an Intel® CoreTM i7-6800K CPU operating at 3.40 GHz with 12 cores, and an NVIDIA GeForce GTX 1080 Ti graphics card.

GPT-3.5

The last one, not directly implemented in the system, was GPT-3.5. The same prompt was tested in this model to compare the performance of the Llama models with this one. The file size of GPT-3.5 is estimated to be around 350 GB. This LLM is capable of understanding and generating text coherently and naturally. The obtained results are analyzed in the Table 5. The therapeutic aspects analyzed in this study include adaptability, quality of feedback, personalization of dialogue, continuity of discourse, and data recording.

Based on these results, we have decided to conduct tests using GPT-3.5 to assess user acceptance. Positive user acceptance would confirm the effectiveness of our approach, justifying further efforts towards the autonomous integration of large language models (LLMs) into the system. Future autonomous implementation may utilize Llama A100, which has also yielded promising results, although its limited memory capacity has currently prevented its deployment.

5.2 Evaluation of the System with Real Users

We tested our complete system with 10 volunteers from our research group using a Likert-scale questionnaire to evaluate various aspects of functionality and per-

Model	Response (seconds)	Time	Dialogue analysis
Llama 3	14.3		This model fully adapts to the participant's level of cogni- tive impairment by tailoring the dialogue accordingly, but its effectiveness could not be correctly assessed due to a lack of memory. Despite this limitation, it demonstrates significant personalization by addressing the participant by name and gender. However, the absence of memory re- sults in no continuity of discourse. It features good data recording capabilities, enabling subsequent evaluation of the therapy administered.
ChatQA-1.5-8B-Q8_0.gguf	33.2		This model does not adapt to the participant's level of cog- nitive impairment, resulting in poor feedback and easily lost conversation threads. It shows minimal personaliza- tion, as it does not address the participant by name. While there is short-term continuity of discourse, the system fails to function properly after three to four messages. Despite these issues, it maintains good data recording capabilities for the subsequent evaluation of the therapy provided.
GPT-3.5	5.1		This model fully adapts to the participant's level of cog- nitive impairment, generating a natural and dynamic con- versation with good feedback. It demonstrates a high de- gree of personalization by addressing the participant by name and gender. The model maintains complete conti- nuity of discourse, even recalling information from past interactions. Additionally, it features good data recording capabilities, allowing for effective subsequent evaluation of the therapy conducted.

Table 5. Comparison of the dialogues generated by each model

formance. Each participant played a selection of the previously described games with EBO, utilizing all the agents in the architecture. Following the gameplay, the survey included questions about user experience, system responsiveness, and overall satisfaction, providing a comprehensive assessment of the system's functionality and effectiveness. The summary of the results from the user satisfaction survey is presented in the Table 6

Question	Average value in the response (standard deviation)
How well did you feel in the interaction with EBO?	4.2 (0.63)
Did you find EBO's instructions and explanations clear and easy to understand?	4.7 (0.48)
Do you feel that the feedback provided by EBO during the game is adequate?	4.1 (0.99)
How easy was it to understand and answer EBO's questions during the game?	4.7 (0.48)
Did you find the questions appropriate for you?	4.7 (0.67)
Do you think EBO's language was appropriate?	4.3 (0.82)
Do you think the interaction with EBO could improve your skills and functions?	4.3 (0.67)
Would you recommend EBO activities and games to others?	4.8 (0.42)
Did you feel that EBO understood all your answers? (Speech recognition)	3.9 (1.1)

Table 6. Results of the interaction with EBO

The evaluation results of the interaction with EBO indicate a predominantly positive experience among participants. Most questions received high scores, particularly those related to the clarity of instructions and ease of understanding, averaging 4.7 with a low standard deviation of 0.48. This reflects strong consistency and overall clarity in EBO's communication. Additionally, the willingness to recommend EBO activities scored the highest, with an average of 4.8 and a standard deviation of 0.42, indicating high user satisfaction. However, the perception of EBO's understanding of responses was lower, with a score of 3.9 and a higher standard deviation of 1.1, suggesting variability and potential issues with voice recognition, especially for male voices. Overall, while EBO interactions are effective and positive, improvements in voice recognition are needed for a consistently satisfying user experience.

6 Conclusion

In this article, we have delineated the development of a system employing social robots like EBO for interactive cognitive games and therapies bolstered by Large Language Models (LLMs), signifying a promising pathway to address the cognitive exigencies of the elderly populace. By harnessing narrative generation, user response capture, and narration capabilities, personalized interactions are attained and tailored to individual preferences and cognitive capacities. The "therapist-in-loop" approach ensures a collaborative framework, where therapists contribute to initial configurations and autonomous therapy sessions are conducted, culminating in comprehensive summaries for subsequent analysis.

This personalized therapeutic approach seeks to heighten user experience and optimize the therapeutic efficacy of social robots in fostering cognitive and emotional well-being among older adults. The pilot study conducted with three different games underscores the potential effectiveness of this approach and indicates a significant stride toward refining cognitive therapies through innovative technological integration. Thus, the integration of LLMs within social robotics holds promise in advancing therapeutic interventions for older adults, representing a pivotal step toward addressing the escalating needs of this demographic group.

Acknowledgment

This work has been partially funded by the PDC2022-133597-C41 project and TED2021-131739-C22 R+D+i projects, supported by Spanish MICIU/AEI/10.13039/ 501100011033 and the European Union's "NextGenerationEU"/PRTR, by the PID2022-137344OB-C31 R+D+i project, supported by Spanish MICIU/AEI/10.13039/ 501100011033 "FEDER Una manera de hacer Europa", "FEDER/UE", and by the FEDER Project 0124_EUROAGE_MAS_4_E (2021-2027 POCTEP Program).

References

 Barata, A.N.: Social robots as a complementary therapy in chronic, progressive diseases. Advances in Experimental Medicine and Biology pp. 105–120 (2019). https://doi.org/10.1007/978-3-030-24230-5 5

- Cifuentes, C.A., Pinto, M.J., Céspedes, N., Munera, M.: Social robots in therapy and care. Preprint (2020). https://doi.org/10.1007/S43154-020-00009-2
- Demange, M., Lenoir, H., Pino, M., Cantegreil-Kallen, I., Rigaud, A.S., Cristancho-Lacroix, V.: Improving well-being in patients with major neurodegenerative disorders: Differential efficacy of brief social robot-based intervention for 3 neuropsychiatric profiles. Clinical Interventions in Aging pp. 1281–1290 (2018). https://doi.org/10.2147/CIA.S152561
- Dinesen, B., Hansen, H.K., Grønborg, G.B., Dyrvig, A.K., Leisted, S.D., Stenstrup, H., Schacksen, C.S., Oestergaard, C.U.: 'hi robot!'-use of a social robot for elderly with dementia: An exploratory study. Preprint (2022). https://doi.org/10.2196/preprints.36505
- García García, J.C., Núñez Trujillo, P.M., Bachiller Burgos, P., Bustos García, P.: Towards the design of efficient and versatile cognitive robotic architecture based on distributed, low-latency working memory. In: Proceedings of IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC 2022). Santa Maria da Faira, Portugal (2022)
- Kalik, J.: Effectiveness of cognitive stimulation personalized by the preexisting cognitive level in older adults. Topics in Geriatric Rehabilitation pp. 78–90 (2022). https://doi.org/10.1097/tgr.00000000000345
- Mahmoudi Asl, A., Molinari Ulate, M., Martin, M.F., van der Roest, H.G.: Methodologies used to study the feasibility, usability, efficacy, and effectiveness of social robots for elderly adults: Scoping review. Journal of Medical Internet Research 24 (2022). https://doi.org/10.2196/37434
- Mendibil Esquisabel, B., Larrad, A., Arrieta, H., Zeberio Etxetxipia, I.: Onkofrail: Personalization of a physical exercise program in older patients with lymphoma. Hematological Oncology pp. 45–50 (2023). https://doi.org/10.1002/hon.3166\ ot33
- Radford, A., Kim, J.W., Xu, T., Brockman, G., McLeavey, C., Sutskever, I.: Robust speech recognition via large-scale weak supervision (2022)
- Rodríguez-Domínguez, M., Bazago-Dómine, M., Jiménez-Palomares, M., et al.: Interaction assessment of a social-care robot in day center patients with mild to moderate cognitive impairment: A pilot study. International Journal of Social Robotics 16, 513–528 (2024). https://doi.org/10.1007/s12369-024-01106-4, https://doi.org/10.1007/s12369-024-01106-4
- Sumioka, H., Torresen, J., Shiomi, M., Chen, L.K., Nakazawa, A.: Editorial: Interaction in robot-assistive elderly care. Frontiers in Robotics and AI 9 (2022). https://doi.org/10.3389/frobt.2022.1020103
- Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G.: Llama: Open and efficient foundation language models (2023)
- Tulsulkar, G., Mishra, N., Thalmann, N.M., Lim, H.E., Lee, M.P., Cheng, S.K.: Can a humanoid social robot stimulate the interactivity of cognitively impaired elderly? a thorough study based on computer vision methods. The Visual Computer pp. 1–12 (2021). https://doi.org/10.1007/S00371-021-02242-Y
- Zhao, W., Yu, X., Qin, Z.: Melotts: High-quality multi-lingual multi-accent textto-speech (2023), https://github.com/myshell-ai/MeloTTS
- Zheng, Z.K., Zhu, J., Fan, J., Sarkar, N.: Design and system validation of rassle: A novel active socially assistive robot for elderly with dementia. Robot and Human Interactive Communication (RO-MAN) pp. 121–126 (2018). https://doi.org/10.1109/ROMAN.2018.8525819

Defining a Standard Blockchain-based Black Box for Autonomous Systems

 $\begin{array}{l} \label{eq:loss} \mbox{Laura Inyesto-Alonso}^{1[0009-0003-4095-700X]}, \mbox{David} \\ \mbox{Sobrín-Hidalgo}^{1[0009-0005-7673-5921]}, \mbox{Claudia} \\ \mbox{Álvarez-Aparicio}^{1[0000-0002-7465-8054]}, \mbox{Adrián} \\ \mbox{Campazas-Vega}^{1[0000-0001-8237-5962]}, \mbox{Ángel M}. \\ \mbox{Guerrero-Higueras}^{1[0000-0001-8277-0700]}, \mbox{ and Vicente} \\ \\ \mbox{Matellán-Olivera}^{1[0000-0001-7844-9658]} \end{array}$

Robotics Group. University of León Campus de Vegazana, s/n, 24071, León (España) {linya,dsobh,calvaa,acamv,am.guerrero,vicente.matellan}@unileon.es

Abstract. Human-system interaction demands a high level of security, especially for those systems that work autonomously. In environments where humans and autonomous systems coexist, safety is crucial because an autonomous system malfunction can put the human's physical integrity at risk. In the same way that an airplane integrates a black box system to record the relevant data during the flight, autonomous machines should integrate their own black box system. Implementing a black box in autonomous systems would help to determine the cause of incidents and assign responsibility after an unexpected event occurs. However, there is currently no consensus on a standard for building black boxes for autonomous physical systems. For this reason, this paper proposes a standard for defining the information that should be recorded by an autonomous system black box. The proposed approach combines this standard with blockchain technology, providing security and tamperresistance to ensure the integrity of the black box data.

Keywords: Accountability \cdot Autonomous Systems \cdot Black Box \cdot Blockchain \cdot Cyber-physical \cdot Standard

1 Introduction

The interaction between humans and autonomous physical systems is becoming increasingly frequent. An autonomous system must independently and reliably achieve goals while adhering to rules, laws, and conventions. It relies on diverse, dynamic, and often imprecise information about its environment, this information can come from external sources, the robot's sensors, or a user's direct instruction [1]. This situation demands a trustworthy relationship between the human and the system. Humans develop higher reliability and trust when they can understand the behavior of the autonomous system [2].

Regulation to certify, explain, and audit autonomous systems is necessary [3]. The European Parliament introduced recommendations on Civil Law Rules on Robotics in January 2017 [4]. These recommendations address liability issues related to the use of robots in public spaces and consider various applications. While they provide a good starting point for addressing these challenges, significant legal gaps remain to be resolved. However, ensuring the reliability of an autonomous system is not an easy task. One of the first steps is to ensure the ability to collect relevant information generated by the system during its execution. The process of gathering and analyzing data in order to find the origin of an action after the execution is called **Accountability**. A robust accountability tool not only helps to understand the behavior of the system but also to track the source of a problem [5]. In an environment where humans and autonomous systems coexist, this capability will be crucial when an accident or incident that involves both, humans and systems, takes place. Therefore, accountability can contribute to the assignment of responsibilities when an incident occurs.

We can find clear examples of accountability features in the aircraft field. Planes incorporate two systems called Flight Data Recorder (FDR) and Cockpit Voice Recorder (CVR) [6]. Both systems are considered as **Black Boxes** (BB). Some vehicles also incorporate their own data recorder system, like the Event Data Recorder (EDR) [7]. This concept is not to be confused with those systems whose internal behaviors are hidden or difficult to understand, but it is about storing relevant system data and ensuring its integrity. For this reason, we propose the following definition for a black box for autonomous cyber-physical systems:

"A module that securely stores the information generated by the system during its functioning."

The information generated during the behavior of an autonomous system is quite complex. The relevant information that should be stored in the black box would be the state of the system, decisions made, hardware-related data, logs generated by the different subsystems, and data for the decision-making process. At this point, the black box could be used to assign responsibility to a user or a subsystem, determine the cause of an incident, or generate explanations about the system's behavior during a specific situation.

It seems evident that every machine that requires a high-security level due to its interaction with humans should incorporate a black box-like system. However, there is no standard agreement regarding the specific information that should be stored in those black boxes. This lack of an implementation of a standard for black boxes integrated into autonomous systems is the main motivation for this work.

Furthermore, it is necessary to highlight that one of the main objectives of a black box is to determine the origin of an unexpected behavior so that responsibilities can be assigned. To reach this objective the black box must be incorruptible to ensure data integrity.

For this reason, one of the keywords used in the proposed definition refers to information security. Blockchain emerges as an interesting approach to solving the problem of building a black box system with robust security features. Blockchain was born as a technology to realize distributed ledgers and has recently attracted significant research attention in the field of cybersecurity. This technology aims to achieve decentralized transaction management, allowing any node in the network to initiate transactions equally according to established rules, without the need for third-party management. Transactions in the system are stored in blocks, which are then linked together in a chain and organized chronologically. Furthermore, transactions recorded in blocks are immutable and transparent to all participants [8].

In terms of data integrity, **Smart Contracts** (SC) serve to guarantee the fulfillment of a transaction between two parts on a blockchain system. They are computer programs stored on the blockchain that are automatically executed when certain conditions are met. SCs can establish rules, like a traditional contract, and enforce them through the code. Therefore, they provide an alternative to ensure successful automatic agreements.

Some works show the possibilities of combining blockchain with black box systems. One example can be found in [9], where the authors propose a black box with anti-tampered features achieved through blockchain technology. One of our previous works also explores the potential of this technology to improve accountability and explainability systems [10]. Integrating blockchain technology into our black box system allows us to ensure the stored information has not been tampered since the system generates it.

This paper proposes a definition of a standard for creating a black box for autonomous systems, capable of storing data securely. In addition, this proposal has been designed to be agnostic to the middleware used by the different systems. This research has two main contributions. On the one hand, we propose a standardization of the information, which is needed for a robust accountability system. On the other hand, we designed a Blockchain-based Black Box, named **BCubed**, a library that uses the mentioned standard and integrates blockchain technology to ensure the integrity of the black box data.

The rest of the paper is organized as follows. First, Section 2 defines the standardization of the black box module analyzing the information needed to be stored. Section 3 details a possible implementation of the standard defined used blockchain technology to ensure the integrity of the data. Finally, our contribution and the next steps foreseen are presented in Section 4.

2 Standard

In this section, we will define the standard for information storage. This standard is targeted at cyber-physical systems that work autonomously. One of the core ideas of this standard is its applicability to different systems, regardless of the software they are built on. In this way, we aim to make the standard useful for a variety of autonomous systems, for example, social robots or autonomous cars. Two systems that use different protocols and middleware to build their behavior.

For the definition of the standard we have based our research on the efforts of Winfield et al. in their work [11], in which they propose a standard draft to build black boxes oriented to social robotics. Our main contribution to the standard consists of its combination with blockchain technology to ensure the integrity of the data stored.

Taking the mentioned draft as the starting point, we have performed some changes to obtain a more agnostic version and make it compatible with blockchain.

2.1 Data organization

In the original work, the authors organize data into three types of **records**: Meta Data (MD), Data Data (DD), and Robot Data (RD). The records are the units of data that will be stored in the black box. MD contains basic information about the robot, DD is used to control the multiple records stored in the black box and RD corresponds to the information obtained through the robot's sensors and actuators.

For our implementation, we propose removing DD, because the integration of blockchain solves the record control issues, and adding a new type of record called Overview Data (OD). We also renamed RD to SD (System Data) because the proposed standard is not only centered on social robots. The new record OD includes a summary of the information stored during the execution and it will be added only at the end. Therefore, the final structure of the data will consist of one MD record at the beginning of the execution session, followed by a series of SD records, and ending with an OD record written at the end of the execution session.

Each record is composed of a set of different values called **fields**. The structure of the MD, SD, and OD records, and their corresponding values, are shown in tables 1, 2, and 3 respectively.

The most important change to the original standard is the substitution of the DD record with the OD record. However, other significant changes were performed. First of all, we remove the checksum field (chkS) for all records because the use of blockchain technology ensures the integrity of the record. Furthermore, all references to robot in field names containing the string "bot" have been replaced with "sys" in order to create an agnostic system.

In the MD record (Table 1) we remove the date and time fields, ebbD and ebbT respectively. We add only a field called recT which includes the timestamp of the record. We also remove the recS field and add two new fields: typR and fieN. The field that refers to the system operator (opeR in the original and opeS in the new proposal) has been moved from the MD to the SD. This is because we assume the operator may change during the execution session.

The SD record (Table 2) has similar modifications. Fields used to control data (typR, fieN, and recT) have been added. The autB field records whether the system is operating autonomously or being teleoperated. The sysX field is of particular interest as it is a user-defined field, which is intended to allow the storage of data with different purposes. This field may be employed for the storage of specific data of the autonomous system, including actions, location, navigation, and so forth, depending on the specific necessities. Alternatively,

Table 1. Structure of Meta Data record. Color red points to the fields removed from the original standard. Color green points to the fields added in the current research. Blue color points that a field was moved from one record to another.

Label	Data	Requirement
recS	record size, field and chars	required
typR	Record Type	required
fieN	number of fields in record	required
recT	timestamp record written	required
ebbD	BB start date record written	required
ebbT	BB start time record written	required
sysN	system name or identificator	required
sysV	system version	optional
sysS	system serial no	optional
sysM	system manufacturer	required
opeS	system operator	optional
resP	name and contact details of responsible person	required
bbnV	black box name and version	required
chkS	checksum for complete record	required

Table 2. Structure of System Data record. Color red points to the fields removed from the original standard. Color green points to the fields added in the current research. Blue color points that a field was moved from one record to another.

Label	Data	Requirement
typR	Record Type	required
fieN	number of fields in record	required
recT	timestamp record written	required
sysT	system time	required
opeS	system operator	optional
autB	autonomous flag	required
actD	actuator no and demand value	optional
actV	actuator no and actual value	optional
batL	battery level	optional
tchS	touch sensor no and value	optional
irSe	infrared sensor no and value	optional
ifSe	line following sensor no and value	optional
gyrV	gyro no and value	optional
accV	accelerometer no and value	optional
tmpV	temperature sensor no and value	optional
micI	microphone no and value	optional
camF	camera no and frame grab	optional
txtC	text input command	optional
txtR	text reply	optional
decC	system decision code and reason	optional
wifi	WiFi status and signal strength	optional
sysX	manufacturer definable field	optional
chkS	checksum for complete record	required

it may be utilized to store log information for the various subsystems of the agent. For illustrative purposes, consider the following examples of sysX fields: {sysX:{01:"My log message"}} or {sysX:{02:"CPU temperature 40°C"}}. Due to the logs stored in these fields, the decC field is no longer necessary and has been removed.

Table 3. Structure of Overview Data record. This entire record is a new addition.

Label	Data	Requirement
typR	Record Type	required
fieN	number of fields in record	required
recT	timestamp record written	required
bbtS	black box total size	required
bbtR	total number of records stored	required

Finally, we create the OD record (Table 3). The purpose of this record is to work as a footer for the black box, pointing to the end of the session. Some of the fields of this record, such as bbtS and bbtR, represent a summary of the information recorded during the execution session.

2.2 Black Box System Design

Taking into account the definition of the standard and the blockchain integration, we elaborate a theoretical approach to the final black box system. Figure ?? shows a diagram that illustrates how the system operates. The process starts when the autonomous system begins the execution of its behavior, generating different types of data.

The black box proposed is a complex system with various software modules that perform different tasks. All of these modules can generate data, which is relevant to the black box. We have named these modules as subsystems. The information of all the subsystems together with the hardware-related data are received as the input of a data curation module, as shown in Figure 1. The data curation module is responsible for controlling data format before it is processed by the blockchain.

Inside the black box system, the API module can be found. This module establishes a channel to exchange data between external agents and the smart contract. The API can be used for explainability or forensic analysis purposes since allows an agent to request certain data stored in the black box. Moreover, the smart contract module is responsible for ensuring the integrity of the data stored and retrieved.

2.3 System Formalization

In this subsection, we propose a formalization of the previous system. This formalization combines the data structures defined in the standard with the blockchain technology into a black box system approximation.



Fig. 1. Design for the proposal framework. Circles denote software modules.

First of all, a data field (f) has been defined as a tuple of two elements: label (l) and value (v); as shown in Equation 1, such that $l \in L$ and $v \in V$. The set of all possible fields is denoted by (F), and defined in Equation 2.

$$f = \langle l, v \rangle \tag{1}$$

$$f_i \in F = \{f_1, f_2, ..., f_n\}, n \in \mathbb{N}$$
(2)

In the system defined exist three types of record, which are a subset of fields. These records are defined as the following types of sets:

- 1. Meta Data or MD. Denoted by M and defined in Equation 3.
- 2. System Data or SD. Denoted by \mathcal{D} and defined in Equation 4.
- 3. Overview Data or OD. Denoted by O and defined in Equation 5.

$$m_i \in M = \{m_0, m_1, ..., m_n\}, n \in \mathbb{N}$$
(3)

$$d_i \in \mathcal{D} = d_0, d_1, \dots, d_n\}, n \in \mathbb{N}$$

$$\tag{4}$$

$$o_i \in O = \{o_0, o_1, ..., o_n\}, n \in \mathbb{N}$$
(5)

It can be established that these different sets are subsets of F, such $M \subseteq F$, $\mathcal{D} \subseteq F$, and $O \subseteq F$. A record can be of any of the defined types as shown in Equation 6. The set of all possible records r_i is defined in Equation 7 and must satisfy the following restrictions: $r_0 \in M$, $\{r_1, r_2, ..., r_{n-1}\} \in \mathcal{D}$, and $r_n \in O$.

$$ri = m_i \lor d_i \lor o_i \tag{6}$$

$$ri \in R = r_0, r_1, \dots, r_n, n \in \mathbb{N} \tag{7}$$

Finally, a function (defined in Equation 8) is applied to transform the records into a blockchain structure. The black box system is denoted by B and defined in Equation 9.

$$bcFunc: R \to B$$
 (8)

$$bi \in B = b_0, b_1, \dots, b_n, n \in \mathbb{N}$$

$$\tag{9}$$

3 Black Box Development Framework Level

This section introduces a Blockchain-based Black Box development framework proposal as a preliminary feasibility study for the proposed standard. The primary objective is to determine the viability of implementing a prototype standard.

To test the standard, we have developed *BCubed*, a library that enables the creation of a standard BB based on blockchain technology, as well as the interaction with this system. The interaction includes the storage of different records, the retrieval of records by type, and the retrieval of all records. To interact with the blockchain technology, a Smart Contract has been implemented in the system to ensure that all transactions are fulfilled. These transactions contain a set of records, whose data are validated by the SC functions before being stored in the blockchain.

In order to present the development in a structured way, it is first necessary to explain the data organization and then to expose the components of *BCubed*.

3.1 Data organization

This subsection presents the technical implementation of the data organization proposed in Subsection 2.1. Two main types of data have been created to represent the aforementioned records: SC structures and API structures. The two types of data represent two distinct versions of the records, named BaseRecord, MetaDataRecord, SystemDataRecord and OverviewDataRecord.

In order to provide a foundation for the implementation, the conceptualization of the SC is developed in the Solidity programming language ¹, while the API module is implemented in Python. Listings 1.1 and 1.2 represent the RecordType enumerations for both modules, SC and API. Listings 1.3 to 1.10 provide a comprehensive comparison between the key SC structures and API structures. Odd listings contain the SC structures, while even listings represent the API structures.

¹ https://soliditylang.org/

```
class RecordType (str, Enum):
enum RecordType {
    NOT_DEFINED,
    MD,
                                                 NOT_DEFINED = 0,
    SD.
                                                MD = 1,
SD = 2,
    ΟD
                                                OD = 3
}
    Listing 1.1. SC - RecordType.
                                                Listing 1.2. API - RecordType.
struct BaseRecord {
    RecordType
                               typR;
                                                 'typR': RecordType.NOT_DEFINED,
    uint8
                               fieN;
                                                'fieN': 0,
'recT': 0
    int256
                               recT:
3
    Listing 1.3. SC - BaseRecord.
                                                 Listing 1.4. API - BaseRecord.
struct MetaDataRecord {
                                            ſ
    BaseRecord
                               base;
                                                 'base': BaseRecord(),
    string
                               sysN;
                                                 'sysN': ""
'sysV': ""
                               sysV;
    string
                                                 'sysS':
                                                         .....
    string
                               sysS;
    string
                               sysM;
                                                 'sysM': "",
                                                 'resP': "",
    string
                               resP;
                                                 'bbnV': ""
    string
                               bbnV:
}
     Listing 1.5. SC - MD record.
                                                 Listing 1.6. API - MD record.
struct SystemDataRecord {
    BaseRecord
                               base;
                                                 'base': BaseRecord(),
                                                'sysT': 0,
'opeS': ""
    uint256
                               sysT;
    string
                               opeS;
                                                 'autB': False,
                               autB;
    bool
    IdUint16ValueInt24
                               actD;
                                                 'actD': IdUint16ValueInt24(),
    IdUint16ValueInt24
                               actV:
                                                 'actV': IdUint16ValueInt24(),
    mint24
                               batL:
                                                 'batL': 0.
                                                 'tchS': IdUint8ValueUint16(),
    IdUint8ValueUint16
                               tchS:
    IdUint8ValueUint16
                                                 'irSe': IdUint8ValueUint16(),
                               irSe;
    IdUint8ValueUint16
                               ifSe;
                                                 'ifSe': IdUint8ValueUint16(),
    IdUint8ValueUintArr16
                               gyrV;
                                                 'gyrV':IdUint8ValueArrayUint16(),
                                                >'accV':IdUint8ValueArrayUint16(),
'tmpV': IdUint8ValueInt16(),
    IdUint8ValueUintArr16
                               accV:
    IdUint8ValueInt16
                               tmpV;
    IdUint8ValueString
                                                 'micI': IdUint8ValueString(),
                               micI;
    IdUint8ValueString
                               camF;
                                                 'camF': IdUint8ValueString(),
                                                 'txtC':
    string
                               txtC:
                                                         .....
                               txtR;
                                                 'txtR':
    string
    IdUint8ValueUint8
                               wifi;
                                                 'wifi': IdUint8ValueUint8(),
    IdUint8ValueString
                               sysX;
                                                 'sysX': IdUint8ValueString()
3
                                                 Listing 1.8. API - SD record.
```

Ł

Listing 1.7. SC - SD record.

struct	OverviewDataRecord	{	
Bas	eRecord		base;
uin	t256		bbtS;
uin	t256		bbtR;
3			

Listing 1.9. SC - OD record.

'base':	BaseRecord(),
'bbtS':	0,
'bbtR':	0

Listing 1.10. API - OD record.

A variety of custom structures are employed for different format values. Thus, the structure IdUint16ValueInt24 is used for values with the format '000:±0000.00', while the structure IdUint8ValueUint16 is utilized for values with the format '00:000'. The structure IdUint8ValueArrayUint16 represents values with the format '00:(±0000:±0000):±0000)'. Meanwhile, the values with the format '00:string' are represented by the structure IdUint8ValueString. Lastly, the structure IdUint8ValueUint8 is employed for values with the format '0:00'.

3.2 Black Box System

Concerning the development of BCubed, three main components can be identified: a Data Curation component that collects information and processes data, but it is outside the scope of this work; the API, which enables interaction with the system; and the Blockchain with the SC implementation, which is responsible for validation and secure storage. Figure 2 depicts a diagram that outlines these components and their primary functionalities.



Fig. 2. Black box system components.

Data Curation. *BCubed* is dependent on a component that is responsible for the extraction of autonomous system information. The primary objective of this component is the collection and subsequent processing of data to obtain the values required for storage. As it was mentioned above, this implementation is limited to *BCubed* system. Therefore, this component is outside of the scope of this development. In the absence of this dependency, the implementation can only be tested through testing suites.

API. This component is responsible for the interaction with the entire *BCubed* system. The API component contains the object responsible for establishing the connection with the blockchain network, compiling a new SC or getting an existing one, as well as interacting with its functions. This is the interface through which the client will communicate. The client is understood to be the component Data Curation mentioned above or an external agent, who wants to verify the *BCubed* records. As previously stated, the client interaction includes both functionalities: the storage and the retrieval of records. Furthermore, the client is responsible for the blockchain server and SC configurations.

Blockchain - SC. On the one hand, to deploy the SC on the blockchain and enable its interaction, it is necessary to have a blockchain local network in place. On the other hand, SC covers the structures explained in Subsection 3.1 and the functions in charge of data validation, data storage, and data retrieval. The high-level structure used in the SC code is named StandardBlackBoxRecords:

struct StandardBlac	ckBoxRecords {
MetaDataRecord	metaDataRecord;
SystemDataRecor	d[] systemDataRecords;
OverviewDataRed	cord overviewDataRecord;
1	

Listing 1.11. SC - StandardBlackBoxRecords

The previous structure contains a single MD record, as many SD records as are required by the system, and a single OD record, which represents all the BB information. Regarding the data validation, there are several stages of verification. Firstly, a series of related record type checks are done to ensure, for instance, that only a single MD record exists; a single OD record exists; if no MD record exists, one must be created to initiate the BB; and if an OD record exists, a new BB must be created because the actual is closed. Secondly, the verification of the potential data to be stored is executed. For example, those values that are required, because they are not optional, must be different from the default values, which include empty strings, zeros, and so forth. These checks manifest themselves as follows:

Listing 1.12. SC - require example

4 Conclusions and Future Works

The main contributions of this work are the definition of a theoretical standard and the subsequent prototype framework *BCubed*, demonstrating that the standard we are based on can be applied in practice and is therefore valuable for building a black box-like accountability system.

Although this research represents a theoretical and initial investigation to achieve a robust and secure accountability system, it is important to note that the first conclusions are encouraging. Nevertheless, further research is necessary to ensure the security and efficiency of the system. As the project progresses, a number of challenges have been identified that will require addressing. These include the following:

- BB lifetime. It is necessary to determine the operational lifetime of the BB to guarantee its correct functioning and utilization. Additionally, it is needed to define a method for the retrieval of BBs that have reached the end of their operational life.
- Data frequency and duplicates. It is important to note that not all data are published with the same frequency. Consequently, some data may be duplicated several times.
- Data curation. As previously stated, a new component is required that collects and processes the autonomous system data in order to populate the structures of the records.
- File storage. The storage of files, such as images or audio, is a challenging process due to the complex and voluminous nature of the data. Moreover, the utilization of blockchain technology may result in economic consequences.
- Improve fields. As the research progresses, a number of new requirements will emerge. It will be important to ascertain whether these necessities are common to different autonomous systems and new fields are required, or whether user-defined data can be included in the sysX field. Consequently, new fields will emerge while others may undergo modification.
- Resources and computational cost. Further research and implementation is required to enable a comprehensive analysis of the resources needed and the computational cost. It is therefore necessary to conduct a detailed examination of these aspects. Additionally, it is important to determine whether the solution will be implemented in an embedded manner or in an isolated hardware.

In light of the aforementioned considerations, it is necessary to identify a solution to the previous challenges and to determine the adaptability of this standard and its applicability to different autonomous systems. To this end, future works will concentrate on testing this standard with robots, particularly on ROS 2², which is the *de facto* standard in the field of service robots.

² https://docs.ros.org/en/foxy/Citations.html

Acknowledgments. This research is part of the project TESCAC, financed by "European Union NextGeneration-EU, the Recovery Plan, Transformation and Resilience, through INCIBE". Moreover, it is part of the Grant PID2021-126592OB-C21 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe". Finally, David Sobrín-Hidalgo thanks the University of León for funding his doctoral studies.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

- P. Abbeel, K. Goldberg, G. Hager, and J. Shah, "Toward a science of autonomy for physical systems: Paths," arXiv preprint arXiv:1609.05814, 2016.
- [2] D. R. Billings, K. E. Schaefer, J. Y. Chen, and P. A. Hancock, "Humanrobot interaction: Developing trust in robots," in *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '12, Boston, Massachusetts, USA: Association for Computing Machinery, 2012, pp. 109–110, ISBN: 9781450310635. DOI: 10. 1145/2157689.2157709. [Online]. Available: https://doi.org/10.1145/ 2157689.2157709.
- T. Keeley, "Auditable policies for autonomous systems (decisional forensics)," Issues for Defence Policymakers, p. 196, 2015.
- [4] E. Parliament, "Report with recommendations to the commission on civil law rules on robotics," 2017, Accessed: 2024-06-10. [Online]. Available: https://www.europarl.europa.eu/doceo/document/A-8-2017-0005_EN.html.
- [5] L. Fernández-Becerra, M. A. González-Santamarta, D. Sobrín-Hidalgo, Á. M. Guerrero-Higueras, F. J. R. Lera, and V. M. Olivera, "Accountability and explainability in robotics: A proof of concept for ros 2- and nav2based mobile robots," in *International Joint Conference 16th International Conference on Computational Intelligence in Security for Information Systems (CISIS 2023) 14th International Conference on EUropean Transnational Education (ICEUTE 2023), P. García Bringas, H. Pérez García, F. J. Martínez de Pisón, et al., Eds., Cham: Springer Nature Switzerland, 2023, pp. 3–13, ISBN: 978-3-031-42519-6.*
- [6] W. Raza, J. Renkhoff, O. Ogirimah, R. S. Stansbury, and H. Song, "Flight data recorders: Past, present, and future," *IEEE Aerospace and Electronic* Systems Magazine, pp. 1–21, 2023. DOI: 10.1109/MAES.2023.3335333.
- [7] J. T. Correia, K. A. Iliadis, E. S. McCarron, M. A. Smolej, B. Hastings, and C. C. Engineers, "Utilizing data from automotive event data recorders," in *Proceedings of the Canadian Multidisciplinary Road Safety Conference* XII, London Ontario, 2001, p. 18.
- [8] Q. Zhou, H. Huang, Z. Zheng, and J. Bian, "Solutions to scalability of blockchain: A survey," *Ieee Access*, vol. 8, pp. 16440–16455, 2020.
- [9] R. White, G. Caiazza, A. Cortesi, Y. I. Cho, and H. I. Christensen, "Black block recorder: Immutable black box logging for robots via blockchain," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3812–3819, 2019. DOI: 10.1109/LRA.2019.2928780.
- [10] L. Fernández-Becerra, M. Á. González-Santamarta, Á. M. Guerrero-Higueras, F. J. Rodríguez-Lera, and V. M. Olivera, *Enhancing trust in autonomous agents: An architecture for accountability and explainability through blockchain and large language models*, 2024. arXiv: 2403.09567 [cs.RO].
- [11] A. F. T. Winfield, A. van Maris, P. Salvini, and M. Jirotka, An ethical black box for social robots: A draft open standard, 2022. arXiv: 2205.06564 [cs.R0].

Upgrading the cognitive architecture of a robot to become socially aware

Jose Galeas¹[0009-0007-9978-8406], Pedro Núñez²[0000-0002-3615-8833], Alberto Tudela¹[0000-0001-6796-5286], Antonio Jerez¹[0009-0006-8187-5301], Óscar Pons¹[0009-0002-6518-0666], and Juan Pedro Bandera¹[0000-0003-3814-0335]

¹ University of Málaga, Málaga, Spain {jgaleas,ajtudela,antoniojcor,opfernandez,jpbandera}@uma.es ² University of Extremadura, Cáceres, Spain {pnuntru}@uex.es

Abstract. Socially Assistive Robots are becoming an important technology to cope with the deep demographic changes of the coming society. These robots are a versatile tool for caregivers and can add value to the services provided in care centers such as retirement homes. However, to meet these expectations, they need to address important challenges. They need to be not only functional, but also acceptable, useful and accessible. They need to adapt their behaviour not only to the physical but also to the social context, adapting their behaviour to the preferences, expectations and needs of the people around them. They also need to explain their behaviour, going beyond data-driven explanations to include causal reasoning. This paper presents the research work carried out in the CAMPERO and SHADOW national projects to implement a socially assistive robot capable of adapting to the social context. This work has been dedicated to the improvement of previous robotic platforms, and also to produce a shared cognitive architecture that includes adaptation mechanisms, different types of memory, and deep multi-agent synchronization. The paper describes the results of functional tests that have been carried out in laboratory environments, before the robot is deployed in a retirement home for a long-term evaluation of the user experience.

Keywords: Social robots \cdot Socially assistive robots \cdot Socially aware robots \cdot Cognitive architectures.

1 Introduction

Service robotics is one of the key technologies of Society 5.0 [6]. Robots able to cooperate with people in daily life environments have become technically feasible, thanks to the recent advances in sensors, embedded processing systems, smart environments, and artificial intelligence (AI), among others. These advances successfully tackle complex tasks such as navigation in daily life environments, object and person recognition, natural language interaction or social context adaptation [9]. Service -and more specifically, social- robots are beginning to be deployed in challenging contexts such as restaurants, hotels, supermarkets,

care centers, or even at home. Regarding these application domains for social robots, it is important to consider the demographic changes that inevitably and dramatically affect the current world population, especially in certain areas such as the United States, Japan, or the European Union. These changes have made Socially Assistive Robotics [3], which focuses on robots able to assist with social interaction, to appear as one of the most challenging and potentially valuable application areas for this new generation of robots [14].

But Socially Assistive Robots (SAR) face important difficulties beyond technical ones. They need to address ethical, cultural and legal issues [16], and consider user needs and preferences throughout all the design process to help in adherence [1,14]. SARs working in sensitive environments, such as retirement homes, need to: (i) be aware of their social context; (ii) be able to adapt its behaviour to this context; and (iii) be able to explain its behaviour, beyond data-driven explanations, considering causal reasoning [12].

This paper focuses on the work addressed in two national research projects, CAMPERO (TED2021-131739B-C21, TED2021-131739B-C22) and SHADOW (PDC2022-133597-C42), that are been carried out simultaneously from December 2022 to December 2024. The research conducted in these projects continue in another national project, INSIGHT (PID2022-137344OB-C3), that started on December 2023 and last until December 2025. The main contribution of this paper is to describe how the robotic platforms and cognitive architecture available at the start of these projects have evolved to incorporate all the elements required to produce socially aware agents. This evolution can be summarized as follows:

- update the employed robotic platforms. The new SARs incorporate advanced sensory and motor capabilities. They also have a different physical aspect, co-designed with end users.
- incorporate long-term memory, to store user profiles, agendas and other meaningful data for the robot operation. These data will be incorporated to the immediate context when required to complement information obtained from direct observation.
- incorporate mechanisms that allow the robot to adapt its behaviour according to the context.
- extend the cognitive architecture used by the robot, to allow communication between different agents.

In order to evaluate the new cognitive architecture, a set of use cases has been co-designed with users (residents and care professionals) from a retirement home. These use cases follow their specific needs and preferences. In addition, the data to be included in the user profiles and agendas have also been defined according to the criteria of these professionals. The resulting cognitive architecture has been integrated into the previous and the new SAR. Then, a functional evaluation was conducted in a controlled environment before the SAR was deployed in the retirement home. This paper describes the evolution of the robotic platform and the cognitive architecture, the experimental setup, and the functional results obtained in this first phase of the evaluation.

2 Starting system

CAMPERO project started two years ago, employing a robotic platform equipped with an instance of a specific cognitive architecture. The platform had been successfully tested in daily life environments in previous research [1,9]. Moreover, the retirement home where the platform has been deployed continues cooperating with the researchers, easing further deployments. In parallel, the SHADOW project aims to develop a proof of concept for a social robot to accompany users in real-world environments. The Shadow robot, developed in this project, has evolved from a Technological Readiness Level (TRL) 2 to TRL 7 and maintained an instance of the same cognitive architecture. The following section details the starting system over which the present solution has been built.

2.1 Robotic platform

The CLARA robot [17] (Fig. 1(left)) is a SAR developed within the CLARC EU project ECHORD++ (FP7-ICT-601116). It was designed to perform Comprehensive Geriatric Assessment (CGA) procedures [1]. In further projects, this role changed, and the platform was tasked with different use cases defined by end-users unrelated to diagnosis or healthcare. Instead, these use cases concerned improving the retirement home's services and relieving caregivers from repetitive, non-value-added tasks such as announcing the menu [9].

The CLARA robot has a differential base from Metralabs GmbH, a tactile screen, speakers, a Sick 2D lidar, and one RGB-D camera. It uses a navigation stack, programmed on the Robot Operating System (ROS) framework [15], to navigate autonomously in unconstrained environments, safely avoiding obstacles. The RGB-D camera complements the lidar sensor for navigation purposes, and allows the detection of objects and people in the environment. The robot also has a safety bumper that stops the robot if it contacts with an obstacle. The robot can use its touch screen to receive input from users. It has a text-to-speech (TTS) module synchronized with a subtitles module. It can, therefore, communicate using speech, images, and subtitles on its screen.

2.2 Cognitive architecture

CLARA robot uses the CORTEX cognitive architecture [2]. This architecture is based on software components in charge of different tasks (monitoring, interfacing, speech generation, battery management, etc.). These components connect to the Deep State Representation (DSR), an inner representation of the immediate context implemented as an oriented graph. The DSR is used as a shared blackboard for all modules to communicate through the so-called *agents*. Fig. 2 shows the CORTEX instance available when the CAMPERO and SHADOW project started two years ago. As depicted, it was mainly programmed using the Robocomp framework [2], although Robot Operating System (ROS) framework [15] was also used for some components, which are connected via dedicated proxies. The system included a Java-based component to control the interfaces shown on the screen. All modules rely on the Ice middleware [7] for communication.

63



Fig. 1. Socially Assistive Robots: CLARA, MORPHIA and Shadow.



Fig. 2. Previous cognitive architecture for CLARA robot (2022).

3 Awareness requirements

The previous system was successfully employed to perform a set of use cases in a retirement home: specifically, the robot was programmed to announce the menu, provide information on demand via its chest screen, and allow relatives to make video calls with the residents. These use cases were performed twice a week for five months. The SAR was able to perform these tasks autonomously.

The evaluation focuses on four indicators: accessibility, usability, social acceptance and user experience, four of the five indicators included in the AUSUS Evaluation Framework ([8]). The indicator "Societal Impact" was not included due to its irrelevance within the project's goals. The behavior of the SAR was perceived positively even after the novelty effect had worn off, and users expressed their willingness to continue using it in the future (see [9] for more details on the results of this evaluation).

The robot did not need to adapt to the users or the context to perform these use cases. This was intentional, as it did not integrate any recognition system. The lack of adaptability led to several issues during the experiments. These issues were mainly identified through interviews with residents and caregivers, and direct observation during the experiments. Some of them are listed below:

- Some residents did not want to interact with the SAR. As the robot could not recognize people, proactive behaviors were forbidden.
- Some residents expected the SAR to recognize them or remember previous interactions. These false expectations disappeared after a few tests, but they negatively affected the experience of these users.
- Even when the robot navigated autonomously, it followed certain paths (e.g., from the charging station to the position where it announced the menu). Use cases were programmed to occur at certain hours. The SAR did not check whether people were present before announcing the menu or whether the called resident followed it when calling for a video call. Thus, the evaluator needed to announce that the robot would perform a use case in practice.
- Using a system based solely on a SAR meant that the robot would always need to be on to update agendas, user profiles, etc. These data had to be moved to an external server to allow evaluators and care professionals to work with them, but this option raised privacy issues.
- Performed use cases were too simple to impact the daily routines of the retirement home. While care professionals understood the potential of the SAR, it did not become a useful tool for them.

Before the CAMPERO and SHADOW projects started, more challenging use cases began to be identified from interviews and focus groups organized with residents and care professionals. Most of them required the SAR to recognize people, be aware of the social context, and modify its behavior according to it. These projects were conceived in response to these identified needs: SHADOW project is a proof of concept for the physical and functional design of a robot capable of following a person in a shared daily life environment; CAMPERO project aims to provide a SAR with an opportunistic planner able to adapt the performed use case, and its parameters, according to the user profile and social context.

CAMPERO project started with activities to define a meaningful set of use cases for the SAR. Thus, a focus group session was conducted to capture the daily routines and interactional situations of both residents and care professionals [11]. The focus group included an initial valorization, an exploration of staff-resident interaction, and a final phase of prioritizing ideal robot uses. After the focus group, the information collected was contrasted against previously explored use cases, and the following set of use cases was finally specified:

- Wandering: This use case allows the SAR to engage users opportunistically. The robot wanders around a certain area, waiting for a perceived context to trigger more meaningful use cases.
- Charging: The robot goes to the charging station if the battery is low. This
 use case was extended by adding a button to the robot's touch screen that
 can be used by authorized users to send the robot to the charging station.
 Thus, these users can effectively send the robot out of the room when they
 feel it is useless.
- Menu selection: The robot approaches a resident and asks them for their menu choices for the coming week. Menu choices are made by touching appropriate images on the touch screen.
- Musical therapy: The SAR goes to the room where the musical therapy activity takes place. There, it plays music and moves around. This use case was included to analyze how the presence of a SAR affects the perception of the activity, even when the robot itself is not interacting with users.
- Reminder: The robot approaches a resident and reminds them of an appointment in their agenda.
- Cognitive therapy: The robot approaches a resident and invites them to perform cognitive exercises on its touch screen.
- Bring me water!: When a person touches a specific button on its touch screen, the SAR recognizes them and offers the possibility of requesting a specific service (e.g., 'someone please bring me water') or calling the care staff. Then, the robot moves to the corridor and communicates the received request once it has located a caregiver.
- Follow me: While the rest of the use cases are specified within the CAMPERO project, this use case is defined and implemented in the SHADOW project. In this use case, the SAR follows a person through the environment.

4 System proposal

A complete update of the existing solution was carried out to create a system capable of performing the previously described use cases. This work involved both the robot's physical and logical structure and, in fact, extended the cognitive architecture beyond the SAR. The new system is described in detail below.

4.1 Robotic platforms

Fig. 1(right) shows the Shadow robot, designed within the project of the same name, as a result of a user-centered design process. The robot meets the requirement specifications gathered from various meetings with experts, addressing both sensory and physical exterior aspects. Currently, Shadow is at a TRL 7, although its deployment in real environments is still limited by the number of available units. Instead, in this paper, we use the MORPHIA robot (Fig. 1(center)), which is a solution specifically implemented by Metralabs GmbH as a socially assistive robot for the elderly. Interestingly, Shadow and MORPHIA robots are very similar, although they have been designed in completely independent co-design processes. MORPHIA robot is equipped with a Metralabs GmbH TORY differential base, a range laser scanner SICK s300 and a circular safety bumper. It also has an Intel RealSense D435i RGB-D camera facing the ground used for navigation, three 2mp Valeo cameras to implement 360^o vision, a Microsoft Azure Kinect RGBD camera also used for perception, and a tablet and speakers for interaction. It has the same text-to-speech (TTS) module employed by the CLARA robot and an updated version of the ROS2 navigation stack 'Nav2' [10]. The images obtained from the Azure camera detect objects from the 80 classes in the COCO database using the YOLOv8 algorithm. These images are also the input of a person recognition module that uses dlib's state-of-the-art face recognition algorithm built with deep learning by Adam Geitgey [5].

4.2 Cognitive architecture

Figure 3 shows the updated version of the CORTEX architecture, currently used in CAMPERO and SHADOW projects. An important difference concerning the previous version is the ability to connect several physical agents (robots, smart environments, etc.) in the same cognitive architecture. This connection is achieved by sharing the DSR. Thus, each physical agent keeps a copy of the DSR (Figure 3 shows two copies, one for the robot, the other for the smart environment), and these copies are synchronized via ROS bridges. If the agents lose connection (e.g., the robot moves through an area without a wireless connection), they can keep working using their local DSRs. When the connection is set again, data are synchronized. Synchronization is achieved by annotating each node and edge in the DSR with a generic attribute detailing the physical agent that made the last change. All updates have an associated timestamp and are only taken into account if the updated timestamp is later than the current one. A physical agent can program tasks to be performed by another physical agent (e.q., if the robot collects the menu data for a person, it programs an action for the smart environment to update those data in the long-term memory).

Another important difference from the previous version of CORTEX is the inclusion of a long-term memory. This memory stores long-term data such as user profiles or the retirement home's agenda. It is implemented as a MongoDB database in an embedded computer integrated into the smart environment.



Fig. 3. Current cognitive architecture for CLARA robot and the Smart Environment (2024).

Regarding social awareness, the most important update in cognitive architecture is the implementation of an adaptation component that modulates the behavior of SAR according to physical and social context data. Thus, when a person is recognized, the node representing them in the DSR is loaded with items that encode their user profile and communication preferences. These items have been discussed with the professional caregivers of the retirement home where the SAR will be deployed and are closely related to the data used by these professionals. User profile items are encoded as Likert scale scores related to physical and cognitive abilities and attitudes toward the robot. Communication preferences determine interface-related parameters (*e.g.*, use of subtitles, font size, or voice volume). The user profile is then employed to adapt the robot's actions to the particular user. For example, changing interaction distance or voice volume.

The adaptation component also considers these user profiles, internal and external physical context, and the agenda of the retirement home to determine which use case the robot should perform. Selecting the appropriate use case can be challenging as the number of possible context configurations is large. Some situations have a straightforward solution (*e.g.*, the robot needs to go to the charging station when the battery level is too low). Others require more complex reasoning. The adaptation component applies an approach based on preference learning [4] in these cases. Preference learning is a sub-field of machine learning that focuses on classification based on preference information, usually provided by experts. In this case, this information was provided by people involved in robot development and retirement home personnel. Several classifiers that score the use cases have been developed using this information. These scores create an ordered list of use cases in which the use case with the highest score comes first.

Figure 4 shows an example of the state stored in the DSR after the SAR picked the menu choices for a user (Oscar). Different node encoding agents (pink), actions (blue), and subsystems (yellow) are employed, including a node encoding the smart environment (named 'brain'). The robot has just finished saying 'thanks!' to Oscar, and the menu choices have been stored as a user profile parameter. The SAR has produced an 'update' action for the smart environment to store these data in the long-term memory. The types of edges and nodes, their attributes, and their evolution during the use case are encoded in an ontology initially developed in CAMPERO and SHADOW projects.

5 Experiments

The evaluation of the proposed solution in the retirement home began in June 2024 and will last until December 2024. Before starting this evaluation in the real context, the system was functionally tested in a controlled environment for two months. The testing environment is a research-living lab, and the users testing the system are the people working in the lab. While not adequate to extract any conclusion regarding user experience, this experimentation allowed to validate the robot's functionality at TRL 4.



Fig. 4. Deep state representation (DSR) of the cognitive architecture, after capturing menu choices for a user.

5.1 Setup

Fig. 5 shows the distribution of the living lab where the robot operates for one hour per day. Following its programmed behavior, it will wander around the place, looking for opportunities to trigger meaningful use cases.

A fictional set of menu choices and agendas was created. Users were also enrolled in the system with fictitious user profiles. While the values loaded into these profiles were not real, the attributes were created according to the profiles used by the care professionals of the retirement home where the SAR is deployed. Table 1 shows these attributes and how they influence robot behavior. To fully test the robot's functionality, the values of these attributes were set for different users to cover all possible circumstances.



Fig. 5. Living lab where functional evaluation has been conducted.

5.2 Discussion

After two months of functional testing, the SAR can perform the use cases as required. Its behavior changes according to the user profile, and the performed use case switches to adapt to different contexts. The robot will therefore be deployed in the retirement home on 17 June, to start the long-term evaluation at TRL 7 (prototype demonstration in real environment).

Some interesting issues were detected in the controlled tests. For example, when the robot recognizes a user, it performs as many use cases as possible to maximize its operational efficiency. This behavior could be annoying, and after some tests in the retirement home, it could be interesting to make the robot decide to save some use cases for later interactions, especially for certain users (e.g., those with higher fatigue levels). The voice type and volume, interaction distance, etc., will also be adapted to each user in the retirement home beyond the initial profiles provided by the care staff.

Communication parameter	Influence on robot behaviour		
Willingness to interact	Whether the robot addresses the person in social interactions or not		
Voice volume	The robot changes its voice volume accordingly		
Subtitles	Whether the robot shows subtitles on its screen or not		
Font size	The robot changes the font size of text shown on its screen accordingly		
Safe distance	Preferred interaction distance		
Enrolled in cognitive therapy	Whether the user is or not in that therapy group		
Reminded	If the person has been reminded about their next activity,		
	no repeated reminders are produced		
Menu choices	Similar to the reminder, but regarding menu choices		
Skill parameter	Influence on robot behaviour		
Visual acuity			
Cognitive capacity			
Motor capacity	Informative parameters that do not directly modulate SAP behaviour		
Mood (baseline)	mormative parameters that do not directly modulate SAR behaviour		
Hearing level			
Fatigue level			

Table 1. User profile attributes and their effect on the robot behavior.

The face recognition system has been tuned to avoid false positives: the key requirement is not to misidentify a person not participating in the experiment. Regular testing in the retirement home will help to adjust this subsystem further. Moreover, as recognition is based on face recognition, it may be necessary to update the robot's behaviors and let it ask a person about their identity if in doubt or call a resident to look at the robot. If necessary, these updates must be made carefully to avoid unwanted or annoying effects.

The tactile screen may be difficult to use for elderly people. The first pilot tests in the retirement home show that alternative interfaces should be explored to improve the accessibility of the SAR. Following previous experience of the research group [13], a button-based interface will be added to the robot.

6 Conclusions and Future Work

The proposed system represents a relevant update to the previous system. It allows the robot to decide which use case to perform and how to perform it according to the social context and the particular person interacting with the robot. It also paves the way for multi-agent systems, as it provides a cognitive architecture that can be directly shared by different agents, whether these agents are mobile robots or smart environments.

Experiments carried out in controlled environments show that the system is feasible from a functional point of view. Therefore, long-term experiments were conducted in retirement homes, where these use cases were defined. The evaluation will now focus on user experience parameters. *Will residents and caregivers accept and use a SAR that performs customized use cases defined according to their needs?* The main objective of the CAMPERO and SHADOW projects, from a social point of view, will be to answer this question once the novelty effect warns off. The authors' research groups are already initiating further projects where adaptation goes a step further. Thus, the SAR should be provided with a mechanism to adapt its behavior and explain it. These explanations should not only be data-driven but also detail causal relationships and provide counterfactual explanations. Having the robot discover and explain these relationships is the next step towards robots that are fully aware and integrated into their social context.

Acknowledgments. This work has been supported by grants PDC2022-133597-C42, TED2021-131739B-C21, TED2021-131739B-C22 and PID2022-137344OB-C32, funded by MCIN / AEI / 10.13039 / 501100011033 and by the European Union NextGenerationEU/PRTR (for the first two grants), and "ERDF A way of making Europe" (for the third grant). The authors would like to thank the residents and professionals of Vitalia Teatinos retirement home for their invaluable and kind cooperation in the design and testing of robot functionalities.

Disclosure of Interests. The authors have no competing interests to declare relevant to this article's content.

References

- Bandera, A., Bandera, J.P., Bustos, P., Calderita, L., Dueñas, A., Fernández, F., Fuentetaja, R., Olaya, A., García-Polo, F., González Dorado, J.C., Iglesias, A., Manso, L., Marfil, R., Pulido, J.C., Reuther, C., Romero-Garcés, A., Suárez, C.: Clarc: a robotic architecture for comprehensive geriatric assessment. In: Proc. of the XVII Workshop of physical agents (june 2016)
- Bustos, P., Manso, L., Bandera, A., Bandera, J.P., García-Varea, I., Martínez-Gómez, J.: The cortex cognitive robotics architecture: Use cases. Cognitive Systems Research 55, 107–123 (2019)
- Feil-Seifer, D., Mataric, M.: Defining socially assistive robotics. In: 2005 IEEE C9th Int. Conf. on Rehabilitation Robotics. pp. 465–468 (July 2005)
- Fürnkranz, J., Hüllermeier, E.: Pairwise preference learning and ranking. In: Machine Learning: ECML 2003. pp. 145–156. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
- 5. Geitgey, A.: Face recognition (2018), https://github.com/ageitgey/face recognition
- Gladden, M.: Who will be the members of society 5.0? towards an anthropology of technologically posthumanized future societies. Soc. Sci. 8(5), 148 (2019). https://doi.org/https://doi.org/10.3390/socsci8050148
- Henning, M.: A new approach to object-oriented middleware. IEEE Internet Computing 8, 66–75 (2004)
- Iglesias, A., García, J., García-Olaya, A., Fuentetaja, R., Fernández, F., Romero-Garcés, A., Marfil, R., Bandera, A., Ting, K.L.H., Voilmy, D., Dueñas, A., Suárez-Mejías, C.: Extending the evaluation of social assistive robots with accessibility indicators: The ausus evaluation framework. IEEE Transactions on Human-Machine Systems 51(6), 601–612 (2021). https://doi.org/10.1109/THMS.2021.3112976
- Iglesias, A., Viciana, R., Pérez-Lorenzo, J.M., Ting, K.L.H., Tudela, A., Marfil, R., Qbilat, M., Hurtado, A., Jerez, A., Bandera, J.P.: The town crier: A use-case design and implementation for a socially assistive robot in retirement homes. Robotics 13(4), 61 (2024). https://doi.org/https://doi.org/10.3390/robotics13040061

- Macenski, S., Martin, F., White, R., Ginés Clavero, J.: The marathon 2: A navigation system. In: 2020 IEEE/RSJ IROS (2020)
- O'Donovan, C.: Accountability and neglect in uk social care innovation. International Journal of Care and Caring 7(1), 67–90 (2023). https://doi.org/10.1332/239788221X16613769194393
- Pearl, J. and Mackenzie, D.: The Book of Why. Basic Books, New York, NJ, USA (2018)
- Romero-Garcés, A., Bandera, J., Marfil, R., González-García, M., Bandera, A.: Clara: Building a socially assistive robot to interact with elderly people. Designs 6(6), 125 (2022). https://doi.org/https://doi.org/10.3390/designs6060125
- Seibt, J., Damholdt, M.F., Vestergaard, C.: Integrative social robotics, value-driven design, and transdisciplinarity. Interaction Studies 21(1), 111–144 (2020)
- 15. Stanford Artificial Intelligence Laboratory et al.: Robotic operating system, https://www.ros.org
- Tian, L., Oviatt, S.: A taxonomy of social errors in human-robot interaction. ACM Trans. Hum.-Robot Interact. 10, 1–32 (2021)
- Voilmy, D., Suarez, C., Romero-Garcés, A., Reuther, C., Pulido, J., Marfil, R., Manso, L., Lan Hing Ting, K., Iglesias, A., González, J., García, J., García Olaya, A., Fuentetaja, R., Fernández, F., Dueñas, A., Calderita, L., Bustos, P., Barile, T., Bandera, J.P., Bandera, A.: CLARC: A cognitive robot for helping geriatric doctors in real scenarios. In: ROBOT (1). Advances in Intelligent Systems and Computing, vol. 693, pp. 403–414. Springer (2017)

Formalizing a Social Navigation Use Case in PDDL

Camino Rodríguez, Alba Gragera, José Galeas, Pablo Bustos, Ángel García-Olaya, and Fernando Fernández

Computer Science and Engineering Department, Universidad Carlos III de Madrid, Spain

camrodri@pa.uc3m.es, {ffernand, agolaya}@inf.uc3m.es, agragera@pa.uc3m.es

Abstract. Social Autonomous Robotics is characterized by intensive and continuous interaction with humans. The use of Automated Planning (AP) within a control architecture has been proposed to define and control the actions of robotic platforms in these environments. While AP enables high-level and declarative specification of robot behavior, creating these models is a time-consuming task that requires an extensive knowledge acquisition process to define them accurately. That is why AP is not widely used in this field, especially for tasks like social navigation, which require addressing a variety of challenges. This paper proposes using AP to model social navigation, addressing potential events typical scenarios may pose. To ease this process, we use an interface for graphically modeling the use case with state transition diagrams, which are automatically translated into the corresponding formalization code. We demonstrate that the resulting model can generate sequences of actions that adhere to expected social navigation and effectively manage potential interruptions in normal execution.

Keywords: Social Autonomous Robotics · Automated Planning · Use Case Modelling

1 Introduction

Social Autonomous Robotics (SAR) [2] deals with robots operating in public spaces. A key area of research in this field is how robots can adapt their behavior based on the sensor data they collect to exhibit flexible yet robust behaviors in such dynamic environments. Automated Planning (AP) [4] has been used in the literature to endow robots with autonomous behavior by using a problem solver, namely a planner, and a control architecture: the planner creates the plan of actions to be performed, while the control architecture manages execution and monitoring. However, developing autonomous systems using Automated Planning (AP) involves creating a formal model, typically using the Planning Domain Definition Language (PDDL), that captures the dynamics of the world and the current scenario to be solved. This process is time-consuming and error-prone, making it one of the main bottlenecks in the development of any AP applications. That is why many social robotic systems are developed using other techniques, especially for tasks like navigation, which require addressing a variety of engineering and human factor challenges [10].

In this paper, we propose using Automated Planning (AP) to implement a social navigation use case, planning both the actions to perform during normal execution and the actions to address possible unexpected events in the scenario. These events could include challenges such as losing track of a person or encountering obstacles

To lower the barrier that the PDDL language might pose, we use an interface specially developed to graphically model social robotic scenarios using transition diagrams [5].

These diagrams depict the states and actions the robot should traverse during execution. This conceptual model of the use case is automatically translated into the corresponding PDDL code, allowing the planner to return the sequence of actions that the robot should perform. This graphical interface simplifies the process, making it accessible for individuals with limited knowledge of PDDL to write PDDL domains. Integrated into a control architecture, this gives the robot the capability to follow a plan for normal execution and replan from a state that deviates from the expected one, allowing it to return to normal execution. The proposed use case was tested in simulation by generating different scenarios. We demonstrated that we created valid domains and problems, resulting in successful plans for navigation. This shows that social navigation can be modeled at a high level using AP, reducing the effort required to model these use cases and easily addressing the challenges posed by human factors.

This paper is structured as follows: Section 2 provides a comprehensive review of the current state of the art in social robotics systems and methodologies for social navigation. Section 3 offers background information necessary to understand the technical components employed in this work. Section 4 presents the conceptual model of the social navigation use case, detailing the transition diagrams used for graphical representation and their translation into PDDL code using the interface. Section 5 delves into the formal model of the social navigation use case, the PDDL domain. Section 6 outlines the evaluation methodology and presents results from simulation-based testing across various scenarios. Finally, Section 7 concludes the paper with a summary of findings, and future work to be done.

2 State of the Art

Social navigation for robots has significantly advanced with reactive and proactive strategies designed for dynamic human environments. Initially, reactive approaches tackled challenges like fixed obstacle speeds, leading to proactive methods that blend human modeling and cooperative planning [6]. These methods aim to improve how robots navigate efficiently and interact socially by considering social norms and adjusting to dynamic human interactions. Recent developments include real-time path optimization algorithms such as the social elastic band [11]. These algorithms distinguish between stationary objects and human presence, adjusting navigation paths in real-time to ensure socially acceptable interactions. They use prediction and anticipation to adapt to changes in the environment, demonstrating effectiveness in simulations and real-world tests across different scenarios.

However, challenges remain, especially in crowded areas where computational demands and the unpredictable nature of human behaviors pose significant obstacles. Addressing these challenges requires further advancements in predictive algorithms and adaptive strategies to enhance the reliability of robots in dynamic social settings. Future research aims to refine social acceptance and navigational adaptability across diverse robotic platforms, facilitating intelligent navigation and seamless interaction in human environments.

Related to the knowledge engineering process behind the development of AP models for Social Robotics systems, various tools have been introduced to support the implementation of such planning domains. In addition to some number of PDDL editors¹ which require deep knowledge about the specification language, we can find in literature systems characterized by automatically translating the resulting visual model into its PDDL formalization [12][7][13]. Although all of these systems use different graphical representations to specify planning domains, they focus on users with a deep knowledge on software engineering and become unmanageable for large domains.

3 Background

A Classical Planning [4] task consists of driving a system from a given initial state to a state where predefined goals are achieved, by applying deterministic actions with known effects [3]. We use the first-order (lifted) planning formalism. We consider classical planning tasks extended with negative preconditions. A planning task is a pair $\Pi = \langle D, I \rangle$, where D is the planning domain and I is a problem instance. A planning domain is a tuple $D = \langle O, P, A \rangle$; where O is a set of *objects*; P is a set of *predicates*, each associated with free variables to form an *atom*; and \mathcal{A} is a set of *action schemas*. An atom is grounded if its arguments do not contain free variables. An action schema $a \in \mathcal{A}$ is a tuple $a = \langle name(a), par(a), pre(a), add(a), del(a) \rangle$, where name(a) is the action name; par(a) is a finite set of typed variables defining the action parameters; pre(a) = $\langle pre^+(a), pre^-(a) \rangle$, where $pre^+(a)$ and $pre^-(a)$ are sets of atoms representing the positive and negative preconditions for the action; and add(a) and del(a) are sets of atoms defining the *positive* and *negative effects* of the action. A problem instance is a tuple $I = \langle \mathcal{O}, \mathcal{I}, \mathcal{G} \rangle$, where \mathcal{O} is a set of typed constants representing problem-specific *objects*; \mathcal{I} is the set of ground atoms in the *initial state*; and finally, \mathcal{G} is the set of ground atoms defining the *qoals*.

Grounded actions \underline{a} are obtained from action schemas a by substituting the free variables in the parameters by constants of the same type in \mathcal{O} . A grounded

¹ http://editor.planning.domains/

action \underline{a} is applicable in a state s if $pre^+(\underline{a}) \subseteq s$ and $pre^-(\underline{a}) \cap s = \emptyset$. Applying a grounded action to s results in $s' = (s \setminus del(\underline{a})) \cup add(\underline{a})$. A plan π is a sequence of grounded actions $\underline{a}_1, \ldots, \underline{a}_n$ such that each \underline{a}_i is applicable to the state s_{i-1} generated by applying $\underline{a}_1, \ldots, \underline{a}_{i-1}$ to $\mathcal{I}; \underline{a}_1$ is applicable in \mathcal{I} ; and the consecutive application of all actions in the plan generates a state s_n containing the goals, $\mathcal{G} \subseteq s_n$.

Classical planning assumes deterministic and known action effects, without external events disrupting the plan. However, real-world environments may involve action failures and changes induced by other agents. Although there exist AP paradigms that consider such non-determinism, a common strategy is to address inherent world uncertainty through replanning: if the state changes and the current plan cannot be executed, a new plan is generated based on the current state [14, 3]. A planning approach with replanning upon failure requires monitoring and execution control architectures. Examples of them are PELEA [1] and PlanSys2 [9], both using Classical Planning [4]. Each high-level action in the plan is translated into low-level commands and sent to the robotic platform, assuming that no events will interrupt the execution. To verify that the plan progresses as expected, external environmental information is gathered from sensors and translated into high-level data to confirm that the expected state aligns with the observed state. If discrepancies are found, the current plan may no longer be valid, requiring a replanning to obtain a new plan to manage the current situation.

4 Conceptual Model of the Social Navigation Use Case

In this section, we will describe the social navigation use case at a high level, accompanied by the corresponding conceptual model. This model will serve as the basis for the subsequent formalization in PDDL. We further divide the explanation of the use case into two aspects: the nominal behavior, representing the ideal execution of the robot where the use case concludes without incidents, and the exogenous events that have the potential to disrupt the normal flow of execution and how the robot handles them.

In this use case, the robot's task is to follow a human guide in a social environment. The nominal behavior of the robot should follow these steps: when the robot's button is pressed, the robot asks the person to stand in front of it. Once the person is correctly positioned, the robot detects and identifies them. It then checks its internal database to verify whether the person has permission to use the robot. If the person is authorized to be followed by the robot, it greets them and starts navigating after that person. At the end of the navigation, when the person indicates they want to end, the robot says goodbye, and the process is completed. That is the ideal case and the main workflow represented in Figure 1, where the states expected throughout the use case are represented by boxes, and the actions to perform are depicted by arrows. The states hold the neccesary information for executing the subsequent action, represented by atomic formulas (detailed in Section 5).



Fig. 1. Visual representation of main workflow of the use case.

However, in a dynamic social environment, numerous issues can arise. If the robot is unable to locate, detect, or identify the person, it will terminate the process. Similarly, if the person is identified but does not have the necessary permissions, the robot will halt the process. Additionally, the process can be interrupted and terminated by an external signal at any time, such as the emergence of a more urgent task. At a high level, all these scenarios result in an *end* state, leading to the termination of the use case. However, at a lower level, each of these actions is expected to be translated into a more user-friendly notification, such as "Sorry, you do not have permission", to enhance user understanding.

While navigating the robot may lose track of the person. If the robot loses sight of the individual, cannot see them, or detects multiple people, it will request the person to please reposition themselves, it will verify that it is the same person it was following before, and resume navigation. All of these these scenarios are depicted in Figure 2. They are depicted at the conceptual level for a better understanding of the use case, but in practice, when the robot rechecks the person after losing track, the navigate action it performs is the normal navigate action. This reconnects with the nominal behavior and allows the use case to continue successfully.

The robot may also encounter blockages during its operations. If obstructed by a person or group of people (social blockage), it will notify them that it is blocked, navigate towards the blocking person or blocking group, and ask them to move. Once they have cleared the path, the robot will continue navigating. In the event of encountering a physical obstacle, such as an object blocking its route, the robot will issue a warning and wait until the path is clear before resuming navigation. It is assumed that someone will intervene to clear the path for the robot. Figure 3 illustrates the robot's response to blockages.

The conceptual models described above have been unified into a single model to represent the use case, allowing different tasks to be connected or interleaved to create more complex behaviors. This model has been implemented in the graphical interface as shown in Figure 4, to facilitate the creation and validation



Fig. 2. Visual representation of the different responses the robot has to complications.



Fig. 3. Visual representation of the case when the robot is blocked.

of the PDDL domain and problem files. In the graphical interface, blue boxes represent states and arrows represent the actions that can be performed in each state. Each state is defined by a set of lifted atoms representing the information that must be true to execute the action. The formal representation of each state and action, along with the resulting translation to PDDL code, is the topic of the next section.



Fig. 4. Use case modelled using the graphical interface. Blue boxes are the states with the corresponding predicates that must be true to execute the action, and the edges connecting them are the actions.

5 Formal Model of the Social Navigation Use Case

The graphical interface enables the modeling of the robot's behavior by defining the corresponding planning elements needed to build the domain action model and automatically generates the PDDL domain. Additionally, it allows the creation of problems by defining initial states and goals. By leveraging this tool, we ensure consistency and completeness in the formalization process.

In this section, we elaborate on the steps involved in creating the social navigation use case described earlier, modeled with Automated Planning using the graphical interface, and present the corresponding PDDL domain and problem representations. See Section 3 for further details on the planning elements used to create a PDDL planning task.

Types: Types are used to categorize objects within the planning problem. These categories help define the roles objects can play in actions and predicates. For our use case, only the type **person** was required to identify different individuals. Other common types for navigation use cases include locations or places, but

since the robot follows the person regardless of their destination, these types are not needed.

Predicates: Predicates define properties or conditions that can be either true or false regarding objects in the domain. They are essential for representing the state of the world and establishing relationships between objects to provide useful information during execution. Predicates can include typed variables (**?per - person**) to form atoms, representing specific properties of those predicates in the world state. Some examples of the predicates and atoms defined in the use case are:

- (located ?per person), (detected ?per person) and (identified ?per person) indicate that the person matching the variable ?per is currently located, detected or identified, respectively. There are more aspects to take into account, for example, once the robot identifies a person, the person_detected predicate must hold true until the robot begins to navigate.
- (navigate ?per person) indicates that the robot is currently in navigation, following person ?per, while (end_navigate ?per - person) signifies that the robot has finished navigating after person ?per.
- (external_signal) indicates the presence of an external signal that influences the robot's behavior.
- (stop ?per person) represents that the robot has stopped interacting with person ?per.
- (farewell ?per person) indicates that the robot has said goodbye to person ?per.
- To address interruptions, error signals such as (error_signal_locate), (error_signal_detect) and (error_signal_lost), which become true when the robot detects any of these situations. These predicates allow our PDDL domain to plan responses accordingly.
- (social_blocked) or (physical_blocked) being true indicate that the robot is currently blocked due to social or physical reasons.
- The (final_state) indicates that the robot has reached the final state of interaction. This predicate was specifically created to keep track of terminal states, making it easier to define problems with end goals.

These predicates form the basis for defining and manipulating the state of the environment and the robot's actions within it. They allow the PDDL planner to reason about how to achieve goals and navigate through various scenarios in the social navigation use case.

States: States are represented by a collection of predicates (or atoms) that describe the conditions or properties at a specific point in time, capturing snapshots of the world. In our case, we define various states that represent different stages of the robot's navigation process:

- The start state includes the predicate (button_pressed), indicating that the button has been pressed and the robot is ready to start the process.
- States like person_located, person_detected, and person_identified help us model the flow of actions and add predicates as events occur.
- The navigating state is active while the robot is following person ?per.
- States such as person_no_locate, person_no_detect, person_no_identif, person_no_permit, and person_lost represent interruptions of the use case and include the corresponding error signal predicates indicating that something has gone wrong.
- All interruptions lead to an intermediate *end* state that includes the (stop ?per - person) atom. This state is not the final state; it serves to prompt the robot to announce that the process is stopping. After this, the process transitions to the final_state (illustrated at the bottom of Figure 4).
- The person_regreeted state reached after finding the lost person has the same predicates as person_greeted (the state right before navigating in the main workflow). It allows the robot to resume navigation and re-enter the main workflow.
- The blockage state social_block includes the predicates (social_blocked) and (navigate ?per - person), showing that the robot is blocked by a person when navigating. The same applies to the state physical_block.
- The final state in the main workflow, called farewelled, includes the predicates (farewell ?per - person) and (final_state), representing that the robot has completed its task and bid farewell to person ?per.

The states are all built by grouping relevant predicates to accurately reflect different stages and conditions during the robot's operation. They allow the planner to reason about the necessary conditions a state must satisfy to execute an action in order to achieve the goals of the social navigation use case, and enable the control architecture to compare the expected state after executing an action with the sensed state.

Actions: Actions define the possible transitions between states, representing changes in the state of the world. Actions include:

- Preconditions: Conditions that must be true in the current state for the action to be executed.
- Effects: Deterministic changes that occur in the world state once the action is executed.

They are high-level operators executed by the robot to achieve specific goals and make changes in the environment. By using these operators, facts are added or removed, creating new situations that the robot must manage. For our social navigation use case, we defined various actions to capture the robot's behavior:

- The action start_navigate has as preconditions (detected ?p), (greeted ?p),(not (social_blocked)) and (not (physical_blocked)) because to start navigating, the person must be both detected and greeted, and the

path must be unblocked. This action is part of the Figure 4, and Figure 5 illustrates the corresponding PDDL code automatically generated from the diagram, that includes those preconditions. Regarding the effects, these are not derived from the next state but are explicitly specified during the action definition.

Fig. 5. PDDL code of the start_navigate action.

- Figure 6 illustrates the PDDL code for the action say_foundyou, which is part of the sequence of actions when the robot loses track of the person. In this sequence, the robot has detected the person again and verified their identity. The say_foundyou action results in the person being greeted, transitioning to the person_regreeted state, to then return to the main workflow.

```
(:action say_foundyou
:parameters (?p - person )
:precondition (and
            (not (backwards-action))
            (detected ?p)
            (checked ?p)
            )
:effect (and
        (not (error_signal_lost ?p))
            (greeted ?p)))
```

Fig. 6. PDDL code of the say_foundyou action.

- For a physical block, the robot issues a warning, while for a social block, it requests permission to pass. The final action in both cases is to check for available space. We created two equivalent actions, check_space_physical and check_space_social, which transition from the waiting_for_space state to the unblocked_way state. These actions also negate the blocked predicates, allowing the navigation process to continue. See the PDDL for one of those actions, check_space_physical in Figure 7.

Fig. 7. PDDL code of the check_space_physical action.

We built the social navigation use case through the definition of predicates, states, and actions. This approach enabled us to model the main workflow, interruptions, and blockages. With the help of the graphical tool, we automatically translated our conceptual representation of the use case into the corresponding formalization in the PDDL domain, thereby avoiding the typical loss of information between translation phases.

Once the domain is already defined, creating a problem instance involves defining the objects that belong to the types present in the domain, the initial state of the world and the goals that need to be achieved. The graphical interface also facilitates this definition by creating objects associated with defined types, and by instantiating lifted atoms with these objects in both the initial state and the goals. This is also automatically translated to the corresponding PDDL file. In this way, given the domain and the problem instance, the planner can generate a sequence of actions to transition from the initial state to the goal state, ensuring the robot's behavior aligns with the desired outcomes in the social navigation scenario. An example of a problem in PDDL code is shown in Figure 8.

6 Evaluation

Evaluating the effectiveness of our formalized PDDL domain to generate valid plans for different scenarios is essential for ensuring the reliability of the robot in real-world settings. To achieve this, we created various problem instances to simulate multiple scenarios, designed to challenge both normal navigation and more complex situations involving interruptions and blockages. The high level planning of the actions was performed using the Metric-FF planner [8].

The evaluation is structured around three different situations, we aimed to test: the normal navigation process, navigation with interruptions where the process is terminated, and navigation with obstacles (both social and physical). These cases help identify the strengths and potential weaknesses in the implemented domain. To accomplish this we created problems as explained at the end of section 5. We provide a description of each scenario and a discussion of the results and observations of the PDDL plan.

Case 1: Normal Navigation Process

In this scenario, we evaluate the robot's ability to follow the person without interruptions or blockages, i.e., we are testing the nominal behaviour of the robot. The goal is to ensure that the robot can successfully complete the navigation task from start to finish under ideal conditions. To achieve this, we created a problem instance with an initial state containing the predicate button_pressed, and whose goal is to achieve the predicate final_state. The syntax of the problem is illustrated in Figure 8.

0:	ASK_LOCATE BOB
1:	DETECT_PERSON BOB
2:	IDENTIF_PERSON BOB
3:	CHECK_PERMIT BOB
4:	SAY_HELLO BOB
5:	START_NAVIGATE BOB
6:	CHECK_END BOB
7:	SAY_BYE BOB
	0: 1: 2: 3: 4: 5: 6: 7:

Fig. 8. Problem to test navigation without interruptions or blockages.

Fig. 9. Generated plan for ideal navigation.

Using the Metric-FF planner, we successfully obtained the expected plan for this problem where the robot sequentially performs all necessary actions following main workflow diagram explained earlier, from asking the person to locate themselves to saying goodbye. The generated plan is shown in Figure 9.

Case 2: Navigation With Interruptions

Navigation can be interrupted under several circumstances: when the person does not position when requested to, when the robot cannot detect or identify the person, when the person lacks permission, when the robot loses sight of the person, or when an external signal interrupts the process. In almost all cases, the robot is expected to inform that there is a problem and then terminate the process. For example, if the initial state of our problem includes (button_pressed),

(located Bob), and (error_signal_detect Bob) (indicating that Bob is located but cannot be detected) and the goal is to achieve (final_state), the generated plan is say_no_detect Bob followed by say_process_stopped Bob.

However, the outcome is different when the robot loses the person. The signal indicating a lost person (error_signal_lost) occurs while the robot is navigating, so these two predicates form the initial state to simulate this scenario. The problem to evaluate this interruption is defined as shown in Figure 10. The resulting plan involves the robot asking the person to locate themselves in front of it again, verifying that they are the correct individual, and then resuming navigation. See Figure 11 for the full plan.

```
(define (problem ProblemLost)
(:domain auto)
(:objects
    Bob - person)
(:init
    (navigate Bob)
    (error_signal_lost Bob))
(:goal
                                (final_state)))
```

0: ASK_RELOCATE BOB 1: CHECK_PERSON BOB 2: SAY_FOUNDYOU BOB 3: START_NAVIGATE BOB 4: CHECK_END BOB 5: SAY_BYE BOB

Fig. 10. Problem to test the robot losing the person during navigation.

Fig. 11. Generated plan for the lost person scenario.

Case 3: Navigation With Blockages

This last case examines how the robot handles navigation when encountering blockages. Blockages can be either social, such as a person standing in the way, or physical, such as an object obstructing the path. The robot is expected to recognize these blockages, take appropriate actions to resolve them, and then resume navigation.

For a physical blockage, the initial state includes the grounded atoms (navigate Bob) (to simulate that the robot was navigating) and (physical_blocked). In this scenario, the robot must warn of the blockage, wait until it is cleared, and then continue navigating. The expected plan is to execute the actions say_physical_block Bob, check_space_physical Bob, and start_navigate Bob in that order, which is exactly what the Metric-FF planner generates.

The scenario for a social blockage is slightly more complex. In this case, the initial state includes the atoms (navigate Bob) and (social_blocked). The robot is required to navigate to the blocking person, request space, wait for them to move, and then resume navigation. Therefore, the plan is expected to include actions such as say_social_block Bob, go_to_block, ask_for_space, and start_navigate Bob, demonstrating the robot's ability to handle social blockages effectively. To illustrate this, we present both the problem setup and the plan generated in Figures 12 and 13, respectively.

```
0: SAY_SOCIAL_BLOCK BOB
1: GO_TO_BLOCK
2: ASK_FOR_SPACE
3: CHECK_SPACE_SOCIAL BOB
4: START_NAVIGATE BOB
5: CHECK_END BOB
6: SAY_BYE BOB
```

Fig. 12. Problem setup for social blockage scenario.

Fig. 13. Generated plan for social block situation.

The results from these scenarios highlight the robustness of model in addressing social navigation challenges. In the normal navigation process, the robot demonstrated the ability to follow a person efficiently under ideal conditions. The interruption scenarios showcased the robot's capability to handle unexpected issues by terminating the process or resuming navigation when possible. The blockage scenarios, particularly social blockages, emphasized the robot's proficiency in navigating crowded environments by requesting space and adjusting its path dynamically.

7 Conclusions and Future Work

This paper demonstrates the application of Automated Planning to model a social navigation use case for Social Autonomous Robotics, using a graphical interface to simplify PDDL model creation. By translating state transition diagrams into PDDL code, we streamline the modeling process and lower the knowledge barriers typically associated with AP, especially for complex tasks like social navigation. We propose an AP-based model that not only handles the nominal workflow of a robot following a person but also addresses potential interruptions and blockages. The use case includes predefined states, predicates, and actions that allow the robot to react to changes in the environment. A user-friendly graphical interface aids in modeling the robot's behavior, translating visual state transition diagrams directly into PDDL code. This interface bridges the gap between conceptual modeling and formal planning languages, making planning accessible to people with less knowledge of PDDL syntax and robotic software.

One promising avenue for future development is the integration of a Large Language Model (LLM) capable of translating natural language instructions into PDDL (Planning Domain Definition Language) code. This advancement would further simplify the user experience beyond our current graphical interface, enabling users who are not familiar with planning and PDDL syntax to define their use cases in everyday language. Future efforts should focus on training these models to ensure accurate and efficient translation. The proposed model was tested in simulation across various scenarios. The evaluation showed that the planner could generate valid sequences of actions for different situations. This makes the work ready for integration into a control architecture. Therefore, as future work we want to deploy this model into the final robot operating in a real environment. For that it is necessary to write first the configuration files (*lowToHigh, highToLow* and *durativeConditions*) to be able to integrate the domain into the execution and monitoring architecture. Additionally, real-world testing will be done to validate the practical applicability of this model and identify any areas for improvement.

References

- Alcázar, V., Guzmán, C., Prior, D., Borrajo, D., Castillo, L., Onaindía, E.: PE-LEA: Planning, learning and execution architecture. In: Workshop of Planning and Scheduling (2010)
- Breazeal, C., Dautenhahn, K., Kanda, T.: Social robotics. In: Springer Handbook of Robotics, pp. 1935–1972. Springer Handbooks, Springer (2016)
- Geffner, H., Bonet, B.: A Concise Introduction to Models and Methods for Automated Planning. Synthesis Lectures on Artificial Intelligence and Machine Learning (2013)
- Ghallab, M., Nau, D., Traverso, P.: Automated Planning: Theory & Practice. Elsevier (May 2004)
- Gragera, A., de Mera, C.D., Tudela, A., Cruces, A., Fernández, F., Garcia-Olaya, A.: Towards an easy use case implementation in social robotics (2022)
- Guillén-Ruiz, S., Bandera, J.P., Hidalgo-Paniagua, A., Bandera, A.: Evolution of socially-aware robot navigation. Electronics 12(7), 1570 (2023)
- Hatzi, O., Vrakas, D., Bassiliades, N., Anagnostopoulos, D., Vlahavas, I.P.: A visual programming system for automated problem solving. Expert Syst. Appl. (6), 4611– 4625 (2010)
- Hoffmann, J.: The Metric-FF Planning System: Translating "ignoring delete lists" to numeric state variables. J. Artif. Intell. Res. 20, 291–341 (2003)
- Martín, F., Clavero, J.G., Matellán, V., Rodríguez, F.J.: Plansys2: A planning system framework for ros2. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 9742–9749 (2021). https://doi.org/10.1109/IROS51168.2021.9636544
- Mavrogiannis, C.I., Baldini, F., Wang, A., Zhao, D., Trautman, P., Steinfeld, A., Oh, J.: Core challenges of social robot navigation: A survey. CoRR abs/2103.05668 (2021), https://arxiv.org/abs/2103.05668
- Pérez, G., Zapata-Cornejo, N., Bustos, P., Núñez, P.: Social elastic band with prediction and anticipation: Enhancing real-time path trajectory optimization for socially aware robot navigation. International Journal of Social Robotics pp. 1–23 (2024)
- Simpson, R.M., Kitchin, D.E., McCluskey, T.L.: Planning domain definition using GIPO. Knowledge Eng. Review (2), 117–134 (2007)
- Vaquero, T.S., Silva, J.R., Tonidandel, F., Beck, J.C.: itSIMPLE: towards an integrated design system for real planning applications. Knowledge Eng. Review 28(2), 215–230 (2013)
- Yoon, S.W., Fern, A., Givan, R.: Ff-replan: A baseline for probabilistic planning. In: ICAPS. p. 352. AAAI (2007)

Quadrupeds Robots in Herding: Metrics for Experimental Validation of Animal-Robot Interactions

Beatriz Jové¹, Alexis Gutiérrez¹⁽⁰⁾, Camino Fernández¹⁽⁰⁾, Lidia Sánchez¹⁽⁰⁾, Francisco J. Rodríguez-Lera¹⁽⁰⁾, and Vicente Matellán¹⁽⁰⁾

Grupo de Robótica. Universidad de León, 24007 León, Spain {bjovc,agutf,cferll,lsang,fjrodl,vmato}@unileon.es

Abstract. This paper discusses the evaluation of experiments involving animal-robot interactions. In particular, the paper focuses on the evaluation of interactions between quadruped robots and herds of sheep. This work has been conducted as part of a project aimed at developing a sheepdog robot, where the evaluation of its performance has to be assessed. However, the evaluation of the performance of a robotic sheepdog has been scarcely faced in the literature. There are competitions for biological sheepdogs, but most of these are evaluated by a subjective jury. The main contribution of this paper is the proposal of an objective methodology that could be used to evaluate both biological and robotic shepherd dogs, even automatically. The paper also discusses other lessons learned during the experiments performed involving robots and dogs.

Keywords: Animal - Robot Interaction \cdot Intelligent Robotics \cdot Computer Vision \cdot SELF-AIR project

1 Introduction

Quadruped robots have gathered significant attention in recent years due to their high flexibility, adaptability, and dynamic performance. These attributes allow them to maneuver through various terrains much like their biological counterparts, making them versatile for a wide range of applications [4], including emergency response, military reconnaissance, infrastructure construction, or acting as robotic shepherds [13].

The interaction of robots and animals has been faced in the literature from different perspectives. For instance, Ethology has greatly benefited from the use of robots, as using robotic implementations allows for controlled experimentation to understand animal behavior better. On the other hand, behavior-based robotics [3] have been greatly influenced by Ethology. Even the term "ethorobotics" [12] has been coined to define this field.

Focusing on the herding domain, the application of quadruped robots can be dated back to early projects like the Robot Sheepdog Project [15] in the 1990s, where wheeled robots were used to herd ducks. Modern approaches seek to address the complexities of herding livestock by enhancing robots' context awareness and interaction capabilities. This involves sophisticated sensor systems and machine learning algorithms that enable these robots to perceive and adapt to their environment, thereby improving productivity, animal welfare, and sustainability in livestock farming [8]. This paper focuses on the problem of quadruped robots for herding.

The design and functionalities of quadruped robots in herding applications are rooted in their ability to navigate diverse terrains and interact with livestock in a manner inspired by natural animal movement. Quadruped robots, which feature four legs or limbs, mimic the movement patterns of four-legged animals, allowing them to traverse various terrains efficiently. This versatility makes them ideal for tasks such as herding in remote or rugged landscapes.

Quadruped robots in herding are equipped with advanced sensors and artificial intelligence systems that enable them to perceive their environment and adapt their movements accordingly[5]. However, robotic herding poses significant challenges due to the complexity of the tasks and the need for practical robotto-animal interactions. Two primary obstacles are the lack of suitable robotic herding platforms and efficient herding algorithms for large numbers of animals. Additionally, there is some social and professional reluctance towards the adoption of robotic herding technologies[10].

In addition to quadruped robots, autonomous drones and ground vehicles have been employed for herding tasks. For instance, the Sky Shepherd technology uses drones to muster sheep flocks by responding to animal behavior, aiming to improve animal welfare [11]. Similarly, unmanned ground vehicles have been used to manipulate grazing distribution, with studies showing that cows exhibit evasive responses without significant signs of fear or discomfort [1].

This is another issue that has to be taken into account when evaluating the behavior of the robots in animal environments, "animal welfare". While initial studies suggest that these robots can interact with animals without causing significant distress, ongoing research aims to ensure that their deployment is both effective and ethical [7].

Studies as [9] have shown that animals' initial responses to a technological intervention are followed by lower levels of usage as the product ceases to be new. This "novelty effect" has been identified and discussed in human–computer interaction, and it will also have to be taken into account in the evaluation.

Experimentally validating the performance of robots interacting with biological beings, humans or animals is challenging, and the main problem faced in this paper. Several approaches to this problem can be found in the literature, for example in [14], a method for evaluating animal performance when interacting with technology is presented. In it, an access control prototype is designed to be used by Mobility Assistance dogs to open doors for their owners. To evaluate the usability of the button, a trial was designed where metrics like the time it takes for the dog to open the door, its independence while doing the task (the need of commands from the trainer), the total time the dog touched the prototype, the number of attempts to open the door, and more were recorded. This demonstrates that objective metrics can be applied to measure the performance of an interaction.

When focusing on evaluating the herding capabilities of quadruped robots in herd management, biological sheepdogs should be considered in the first place. The Swedish Sheepdog Society used a standardized method [2], "Herding Trait Characterization (HTC)", to describe herding behavior of individual Border Collies for breeding purposes. The HTC is a non-competitive method to describe how individual dogs typically expressed a number of traits considered important for herding ability. The two versions of this method are based on subjective interpretation of the performance of the dogs by the judges.

In the same way, in Australia, the "Herding Dog Assessment Form–Personality (HDAF-P)" has been proposed [16] to facilitate collection of data on relevant behavioral phenotypes of large numbers of working Kelpies and to apply the HDAF-P to identify personality traits needed for herding dog performance.

The use of video recordings as a main source of information has already been proposed in the literature. In [6], video recordings of herding in a yardbased competition trial are used to identify the most significant interactions. Specialized software was designed to identify behaviors as, for instance, the dog chasing the sheep and a group of sheep escaping the main flock; or a single sheep escaping the flock and the dog chasing it; or more relevant ones for our study as the sheep initiating movement followed by the dog.

The rest of the paper is organized as follows. Section 2 proposes a method (named VBES) for evaluating shepherd dogs, both biological and robotic ones based on video recordings. Section 3 presents two different scenarios where the VBES method has been used. Section 4 discusses the results obtained by VBES in the two scenarios, analyzes statistically the significance of the data, and summarizes lessons learned using the proposed method to a case. Last section presents the conclusions and future work envisioned.

2 Video-Based Evaluation of Sheepdogs (VBES)

This paper proposes a new method to record and evaluate the interactions between a herd and a sheepdog. This evaluation will then be used to compare the herd interaction with a traditional sheepdog and a quadruped robot performing the same task. The aim is to determine whether it is possible to perform this task using technological methods without compromising the results or the welfare of the herd.

The proposed method consists of three stages: video recording, data processing and feature extraction. The first step in this process is the acquisition of videos reflecting the interaction of the herd with both a herding dog (Animal-Animal Interaction) and a quadruped robot (Animal-Robot Interaction). After some consideration, the use of third person videos for evaluation has been deemed the best choice, in particular drone-based, if possible. On one hand, there is a fair amount of videos of this type online, which can be seen with a simple Youtube search. These videos can be obtained from ground cameras, drones, etc. However, not just any video will do for this evaluation. Standards and protocols should be established to try to standardize and facilitate the evaluation.

Even following protocols to reduce the difference in the tests as much as possible, there are other factors such as weather conditions, time of day or even herd mood, which are difficult to monitor and modify and which affect the experiment significantly. Therefore, it is essential to attempt to make the relevant modifications within the available capabilities to reduce the differences between the experiments as much as possible, significantly reducing the impact of other conditions on the tests.

As a consequence, it is important that the videos are taken under the same conditions for both interactions so that the evaluation metrics are as fair and accurate as possible. This means using the same recording equipment, positioned at the same distance and orientation for both sets of interactions. Additionally, the testing area must remain consistent, featuring the same obstacles, to ensure that the tasks performed present the same challenges and advantages in all scenarios.

Regarding feature extracting, the intent is to calculate the distances between detected objects. This can be done in pixels, or, by knowing the real-world dimensions of an object (such as the quadruped robot), the pixel measurements can be converted to another unit. The distances between each detected sheep and the sheepdog or robotic dog can give a sense of how the herd reacts to the different dogs.

Taking into account all of the above, the proposed method is detailed as follows:

- Video recording Data acquisition should be conducted using a flying drone positioned just above the experimental area. By angling the camera directly toward the ground, a 2D view of this area can be captured, which is helpful for measuring objective metrics directly from the images.
- **Data processing** Once the data is acquired, the next step is to process it so as to extract features to help develop metrics and compare the interactions. Using YOLOv8, the sheep, sheepdog and robot can be tracked throughout the frames. This involves not only detecting the objects, but also associating them between frames to identify each object as unique. This means maintaining a consistent identity for every object throughout the video, which allows for accurate tracking of their movements and interactions over time.
- Feature extraction The center of each bounding box is extracted, and the Euclidean distance between the boxes is calculated. This way, the maximum, minimum, and mean distances between the herd and the dog are extracted to facilitate the evaluation. Lastly, the maximum distance between sheep is also recorded for each frame. This metric can provide insight into the compactness of the herd and help indicate if a sheep is straying away from the rest. The object speeds are calculated using the positions of the bounding boxes and the frame rate of the videos. Instead of calculating speeds frame by frame, they are calculated at specific intervals, as objects may not move

significantly between consecutive frames. Similar to distances, the maximum, minimum, and mean speeds of the herd are noted, along with the speed of the dog.

Table 1 summarizes the results obtained after applying the proposed method, which corresponds to the final organization of the dataset's attributes by rows, along with a brief description of each attribute's meaning.

Dataset attribute	Description		
frame	Frame number		
frame_bboxes	List of all bounding boxes in the frame: each one contains		
	the id, class label and coordinates x1, y1, x2 and y2 of the		
	box		
distances	List of all euclidean distances between the detected sheep		
	and the dog in pixels		
min distance	Minimum of 'distances'		
max distance	Maximum of 'distances'		
avg distance	Mean of 'distances'		
furthest_sheep_distance	Maximum euclidean distance between two sheep to give		
	insight into the compactness of the herd		
speeds	List of the speeds of all detected objects in an interval of		
	5 frames obtained by comparing the positions of the		
	objects in the initial and final frame		
dog_speed	Dog speed extracted from 'speeds' for easier data analysis		
min_speed	Minimum of 'speeds'		
max_speed	Maximum of 'speeds'		
avg speed	Mean of 'speeds'		

Table 1: Organization of the extracted data, each row represents a column in the final dataset

To provide further insight into the process, a more detailed explanation is necessary. Once the videos have been acquired, a YOLOv8 model is used to analyze them. Several models have been tried for the different videos and their detections have been analyzed through visual inspection. For instance, the 'large' model was more capable of detecting the quadruped robot compared to the 'extra large', which did not detect anything at all. 'The large' model made some mistakes, such as misclassifying the robot as other categories (motorcycle and zebra). However, these errors can be addressed in the code by identifying which classes are being incorrectly assigned to the robot. Moreover, with the 'large' model, sheep were sometimes detected as birds, but this was also corrected later in the code. This issue could be prevented by creating a dataset for the recognition of the robotic dog, which will be addressed in future work.

For the distance estimation, as the measures used are pixels, every single frame must have the same dimensions. This ensures that the recorded data is not biased. For the speed estimation, the frame rate of the videos must be known. Note that the extracted speed is not measured in m/s, but is the ratio between pixels and number of frames respectively. To ease the workload, all videos have the same frame rate. The frames for this experiment have a resolution of 1920x1080 pixels, and a frame rate of 30 fps.

It is also important to note that while YOLO is great at tracking, objects sometimes go out of frame, so their ID is lost. The sheep also have a natural tendency to go towards the center of the herd, so that they are not exposed at the sides. This behavior flourishes when the sheep face an intruder, much like an unknown sheepdog. When a sheep goes into the core of the herd, its ID is likely to be lost. This is relevant when trying to estimate the speed of the objects, as it is based on the IDs of the objects and their association between frames.

The created dataset has quite a few missing values. This is due to different reasons. If the dog is missing from the frame, then the attributes 'distances', 'min_distance', 'max_distance' and 'avg_distance' will surely be empty. If one or fewer sheep are detected, 'furthest_sheep_distance' will also be empty, as there needs to be at least two sheep to calculate this distance. The speed related attributes can also have missing values, as the speed estimation is performed across a window of frames, in this case five. If the dog or herd are not in one of the two selected frames, the speed cannot be calculated and will be left empty.

The proposed method has been tested in two real scenarios and the whole process is described in the next section.

3 Use cases

As this work is inspired by sheepdog trials and competitions, the tasks to be performed are similar to those found in such events. Some proposed tasks include guiding the herd through a "gate", conducting the herd to a different fenced area or maneuvering the herd inside or outside a "pen". The pen is an improvised structure made out of gates with an opening and, normally, a door to close it.

The proposed method, VBES, has been put to the test with two experiments: a professional sheepdog exhibition and a sheep pen scenario with the robotic sheepdog.

3.1 Sheepdog exhibition

A professional sheepdog exhibition was held during the execution of this project. Its aim was to showcase the natural talents of sheepdogs and the required skills to perform their daily tasks. Figure 1 shows the layout of the course to be completed by the professional sheepdogs. It included two parts: one for testing the dogs' obedience and another for evaluating the dogs' control over the herd.

The first part consisted in a short curved path marked by flags, depicted in the image as red and white circles. The dog must cross between the flags marking the path for this part of the trial to be completed (see Figure 2). The shepherd is constantly shouting or whistling to command the dog to go left, right, stop, come back, etc. The obedience test can be skipped for this experiment, as a


Fig. 1: Layout of the sheepdog trial



Fig. 2: Sample of the first test of the trial: the dog following the marked path

robotic dog is remotely controlled by the shepherd and can hardly be beaten by a traditional sheepdog in this task.

The second section of the test involved the dog approaching the sheep and leading them to the pen (see Figure 3). The dog should be cautious not to get too close to the sheep when establishing first contact, as this could lead to them running away and make the task harder.

Sheep will instinctively run away from the dog, so the dog must approach them slowly to get them inside the pen (see Figure 4). However, the sheep are naturally repelled by the pen, especially if the fences that comprise it are solid or opaque. As the herd realizes it is a dead end, it will avoid going inside.

It is important to note that the shepherd should not move from the assigned position throughout the trial. This is to ensure the dog can work without the shepherd at its side. However, if the sheep become stuck, for example, in a corner,



Fig. 3: Sample of the start of the second test of the trial: the dog approaching the herd



Fig. 4: Sample of the end of a successful trial where the herd is inside the pen

the shepherd can help the dog by making movements and noises or by using a crook.

The optimal way to collect data is using an unmanned aerial vehicle directly above the field with the camera angled downward. This way, all recordings would be at a consistent height above the ground, ensuring a '2D' perspective. Later, when processing the data, images taken from this angle would allow for more precise measurements of distances between detections, as well as more accurate speed calculations.

However, due to weather conditions and flying drone area restrictions, data could not be recorded using the desired method. Instead, videos were recorded with a standard smartphone by a person standing on top of the fence.

The quadruped dog was unable to participate in the competition, so VBES will be applied in this case only to biological sheepdogs.

3.2 Sheep pen scenario

This experiment was carried out in the Experimental $Farm^1$ of the University of León. In this scenario, the sheep had to remain inside the pen because the area's airspace is restricted, preventing outdoor drone flights. The sheep pen has an elongated rectangular shape, so the drone was positioned close to the ceiling at one end so that the videos would capture the clearest view possible (see Figure 5).

This particular herd is unaccustomed to any sheepdog, so their nerves may be more heightened compared to the other herd. However, at the time of these recordings, the herd had interacted with the robot on about four occasions over several months.

Since a fenced "pen" could barely fit inside the sheep pen, and it was previously stated that the obedience test was unnecessary, the intent was to move the herd from one side of the sheep pen to the other and analyze the interactions. As the space is smaller and more elongated than the one in the sheepdog competition, the sheep seemed to feel more enclosed and run away from the robot. The experiment consisted in trying to lead the herd in a calm manner from one point of the sheep pen to the other, repeatedly.



Fig. 5: Sample of the recordings of the robot-herd interaction inside the pen

3.3 Data processing and feature extraction

The proposed method for data extraction was applied to the data collected in the two scenarios mentioned above (see Figure 6). The objects detected by YOLO as 'sheep' are depicted in blue, while the ones detected as 'dog' appear in red.

4 Evaluation

In order to validate the VBES method, a first simple test was to determine whether there are significant differences between the sheepdog exhibition and the

¹ https://centros.unileon.es/veterinaria/granja-experimental/



Fig. 6: Modified YOLOv8 detection output for the sheepdog (left) and the robotic dog (right) experiments. Objects labeled as 'sheep' are depicted in blue, and those labeled as 'dog' in red.

sheep pen scenario. Eight different videos were analyzed. Four of them belong to the professional sheepdog exhibition, their length ranging from 4 to 8 minutes, and the remaining four, to the sheep pen experiment with the robot, with lengths between 2 and 7 minutes. The videos have a resolution of 1920x1080 pixels, and a frame rate of 30 fps. Data from every frame has been extracted, following the attributes depicted in Table 1. In total, the dataset contains 75,323 samples, one for each frame.

Among all the attributes of the dataset, four variables have been selected to be analyzed in order to obtain a general description of the behavior of both herds and both herding methods (real dog vs. robot):

avg distance The mean of distances from the dog to each sheep.

 $\min_distance$ The distance between the dog and the closest sheep of the herd. dog_speed The speed of the dog.

 avg_speed The speed of the herd, as the average of the speed of each sheep in it.

The statistical analysis of the data obtained through VBES shows that it does not follow a normal distribution, so the Mann-Whitney non-parametric statistical test was selected to compare the samples.

This test shows that there is a statistically significant difference between the dog and the robot for the $avg_distance$ indicator ($W = 1.328x10^7; p < .001$), with mean values of 474.722 and 583.758 for the dog and the quadruped robot respectively. $min_distance$ indicator also shows a statistically relevant trend ($W = 1.703x10^7; p < .001$) with means of 322.112 for the sheepdog and 288.216 for the robot.

The analysis of the dog_speed variable shows a statistically significant difference as well ($W = 8.903 \times 10^8$; p < .001), with means of speed of 54.431 and 4.010 for the biological dog and its robotic counterpart respectively. Finally, a statistically meaningful trend has been also observed at the analysis of the herd speed with the avg_speed variable ($W = 6.353 \times 10^8$; p < .001) with means of speed of 71.625 for the herd shepherded by a real dog and 76.908 for the herd shepherded by the robot. The average distance from the dogs to the center of the herd is also different in both cases, being the sheepdog the closest one in general to the sheep, which reveals that the biological dog needs to get closer to the herd to move it in the desired direction. In fact, it is common for sheepdogs to bark, push and even bite the sheep to make the herd follow their instructions. This can be seen in our dataset, which sometimes shows a value of 0 units in the minimum distance between the dog and the nearest sheep. This never happens with the robotic dog, as it never gets close enough to the sheep to touch them.

However, analysis of the data shows that, in general, the robotic dog is closer to the nearest sheep than the real dog. This can be explained by the fact that the herd in the second scenario was enclosed inside a shed and could not move further away even if it wanted to.

The analysis carried out on the variable *dog_speed* shows that the speed of the robotic quadruped is significantly lower than the speed of its biological counterpart, as expected. Although this is a clear disadvantage for the robotic dog compared to the real one, it can actually be compensated by the fact that the robot is remotely operated by the shepherd, whereas the real dog usually needs commands to carry out the assigned task and does not obey immediately.

The last variable analyzed, avg_speed has also shown a significant difference between the behaviors of both herds. In this case the sheep used with the quadruped robot moved faster than the sheep used in the first scenario, which can be explained by the fact that in the second experiment the sheep were constantly moving around the whole area of the sheep pen while the herd used with the biological dog was sometimes stationary. In addition, it was found that the sheep became more frightened at the beginning of the test with the robot, which undoubtedly also increased their average speed during the experiment.

4.1 Lessons Learned

In addition to the conclusions drawn from the statistical analysis, the experiments carried out have also provided some valuable lessons. The goal was to design an objective way of measuring the performance of sheepdogs and robotic dogs equally, for this, at least a trained sheepdog, an expert shepherd, a herd and a quadruped dog are needed. The first lesson is that these elements are quite difficult to obtain.

We did have the opportunity to attend an exhibition show of professional sheepdogs and record the development of the trials. The recorded data from the exhibition was later analyzed. Since the goal is to differentiate between the sheepdog and the herd, YOLOv8 detection was applied. However, several people brought their own dogs with them so, in certain frames, the model was detecting multiple dogs.

The second lesson is that real scenarios are really unpredictable. In particular, to solve this issue, once the bounding boxes of the herd were obtained, their centers were calculated and all dogs that were not the closest to the herd were ignored. Another issue that emerged was the more frequent occurrence of sheep being detected as dogs than anticipated in real scenarios in particular when analyzing the data recorded in the experimental farm with the herd and the robotic dog. However, because the shapes of the herd inside the pen are not as 'circular' as in the competition field — due to the pen's elongated shape and the limited space for the sheep to move freely — adjusting the detected dog to be the one furthest from the herd does not significantly impact the results.

Other issues involve the inability to film the interactions with an unmanned aerial vehicle from above as desired. In spite of this, the designed method should be valid to evaluate data recorded that way, providing more reliable information about the metrics. Furthermore, using a UAV for recording allows for the calculation of actual distances and speeds in units such as meters and meters per second (m/s). This is achieved by knowing the size of an object appearing in the video, such as a quadruped robot or a dog. Since the drone films from a constant position, the object's size in the video remains relatively constant, enabling accurate conversions from pixels to meters. Similarly, by knowing the speed of the video, it is possible to perform the necessary calculations to obtain speeds in meters per second.

5 Conclusions and Further Works

The idea of evaluating the quadruped dog against a regular sheepdog came up, but how could this be made possible? Sheepdog trials appeared to be a promising starting point. They are a way to evaluate the performance of real sheepdogs, so it seemed obvious that would be the right way to do it. However, in these competitions every participant starts with the total score and there is a jury who, somewhat subjectively, deducts points every time the dog does something incorrectly or not perfectly. Since there have not been competitions in the area and time of this investigation, simply participating with the robotic dog in one of them and obtaining a measure of its performance has not been possible.

The main contribution of this paper is the proposal of a method for the performance evaluation of robotic sheepdogs based on video analysis. There are no comparable proposals in the literature for robotic sheepdogs, only for biological ones. A preliminary formalization of the method has been made and tested on recordings of two different environments with real sheep, one involving a real dog (outdoors) and the other one with the robotic dog (indoors).

The data obtained from the automatic analysis of the videos shows that the method is feasible, although significant differences have been found between the biological and the robotic sheepdogs. It must be taken into consideration that an actual sheepdog runs faster than the robotic dog used for this experiment, which can only reach a speed of 1.6 meters per second. Despite this, due to the slow movements of the robot, the sheep seemed not to run away as quickly as they do with a traditional sheepdog.

Some missing values are always expected in datasets that come from a real world experiment, and they do not present a problem. However, too many incomplete samples can slow down the project and lead to inaccurate results. For this specific case, labeling the data for model training would have been beneficial, but time constraints were a limiting factor.

Future works envisioned comprise the extension of the experiments, potentially repeating them with the same herd and in the same area to ensure fairness and consistency in the results. The videos could be recorded using an unmanned aerial vehicle, so that the view is the same for both scenarios and the extracted values for the metrics are more consistent. Ideally, several tests should be designed so that more data can be recorded.

Another possible line of action proposed by livestock experts could be the inclusion of objective measures of herd disturbance in both cases. In this sense, their lactation performance or cortisol levels could be measured. This approach would provide a deeper insight into animal welfare, ensuring that technological applications do not compromise it.

Acknowledgments. This work has been partially funded by Grant TED2021-132356B-100 funded by MCIN/AEI/10.13039/501100011033 and by the "European Union NextGenerationEU/PRTR"

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

- Anzai, H., Kumaishi, M.: Effects of continuous drone herding on behavioral response and spatial distribution of grazing cattle. Applied Animal Behaviour Science 268, 106089 (2023). https://doi.org/https://doi.org/ 10.1016/j.applanim.2023.106089, https://www.sciencedirect.com/science/ article/pii/S0168159123002617
- Arvelius, P., Malm, S., Svartberg, K., Strandberg, E.: Measuring herding behavior in border collie-effect of protocol structure on usefulness for selection. Journal of Veterinary Behavior: Clinical Applications and Research 8, 9–18 (1 2013). https: //doi.org/10.1016/j.jveb.2012.04.007
- Birk, A.: Behavior-based robotics, its scope and its prospects. In: IECON '98. Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (Cat. No.98CH36200). vol. 4, pp. 2180-2185 vol.4 (1998). https://doi. org/10.1109/IECON.1998.724059
- Chai, H., Li, Y., Song, R., Zhang, G., Zhang, Q., Liu, S., Hou, J., Xin, Y., Yuan, M., Zhang, G., Yang, Z.: A survey of the development of quadruped robots: Joint configuration, dynamic locomotion control method and mobile manipulation approach. Biomimetic Intelligence and Robotics 2(1), 100029 (2022). https://doi.org/https://doi.org/10.1016/j.birob.2021.100029, https://www.sciencedirect.com/science/article/pii/S2667379721000292
- Cruz Ulloa, C., Sánchez, L., Del Cerro, J., Barrientos, A.: Deep learning vision system for quadruped robot gait pattern regulation. Biomimetics 8(3) (2023). https://doi.org/10.3390/biomimetics8030289, https://www. mdpi.com/2313-7673/8/3/289

- Early, J., Aalders, J., Arnott, E., Wade, C., McGreevy, P.: Sequential analysis of livestock herding dog and sheep interactions. Animals 10, 352 (2020). https: //doi.org/https://doi.org/10.3390/ani10020352
- Ángel González-Santamarta; José Manuel Gonzalo-Orden; Camino Fernández-Llama; Vicente Matellán; Lida Sánchez., F.J.R.L.M.: Lessons learned in quadruped deployment in livestock farming. In: Workshop on Field Robotics - ICRA 2024. Yokohama, Japan. (2024). https://doi.org/https://doi.org/10.48550/arXiv. 2404.16008
- Herlin, A., Brunberg, E., Hultgren, J., Högberg, N., Rydberg, A., Skarin, A.: Animal welfare implications of digital tools for monitoring and management of cattle and sheep on pasture. Animals 11(3) (2021). https://doi.org/10.3390/ ani11030829, https://www.mdpi.com/2076-2615/11/3/829
- Hirskyj-Douglas, I., Webber, S.: Reflecting on methods in animal computer interaction: Novelty effect and habituation. In: Proceedings of the Eight International Conference on Animal-Computer Interaction. ACI '21, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3493842. 3493893, https://doi.org/10.1145/3493842.3493893
- 10. James Vincent, The Verge, .m.:: A robot sheepdog? no one wants this, says one shepherd, accessed: 2024
- Li, X., Huang, H., Savkin, A.V., Zhang, J.: Robotic herding of farm animals using a network of barking aerial drones. Drones 6(2) (2022). https://doi.org/10.3390/ drones6020029, https://www.mdpi.com/2504-446X/6/2/29
- Miklósi, A., Korondi, P., Matellán, V., Gácsi, M.: Ethorobotics: A new approach to human-robot relationship. Frontiers in Psychology 958 (2017). https://doi. org/https://doi.org/10.3389/fpsyg.2017.00958
- Robotics Group, U.d.L.: Self-air project website. https://robotica.unileon.es/ projects/shephered_robot/ (2022), accessed: 2024
- Ruge, L., Mancini, C.: A method for evaluating animal usability (meau). In: Proceedings of the Sixth International Conference on Animal-Computer Interaction (2019). https://doi.org/10.1145/3371049.3371060, https://doi.org/10. 1145/3371049.3371060
- Vaughan, R., Sumpter, N., Frost, A., Cameron, S.: Robot Sheepdog Project Achieves Automatic Flock Control. In: From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior. The MIT Press (07 1998). https://doi.org/10.7551/mitpress/3119.003.0074, https://doi.org/10.7551/mitpress/3119.003.0074
- Wilson, B.J., Arnott, E.R., Early, J.B., Wade, C.M., McGreevy, P.D.: Valued personality traits in livestock herding kelpies—development and application of a livestock herding dog assessment form. PLoS ONE 17 (4 2022). https://doi.org/ 10.1371/journal.pone.0267266

Graph-based inner state prediction for a socially aware robot

José Tejón-Moreno¹, Juan Pedro Bandera¹, Antonio Bandera¹, and Adrián Romero-Garcés¹

University of Málaga, Málaga, Spain {jtejon,jpbandera,ajbandera,argarces}@uma.es

Abstract. Robots working and interacting with people in daily life scenarios require to be aware of their context, both physical and social. Predictive capabilities allow the robot to adapt to its context, and are a key component of awareness. This paper presents a system that uses the current and past context knowledge to infer future context. This system is integrated into CORTEX, a cognitive architecture where immediate context is represented as an oriented graph, the Deep State Representation. The proposed approach is based on collecting and classifying all the information obtained from the executions of use cases performed in the past by the robot. Then it uses these data to train a Graph Neural Network to predict the next configuration of the Deep State Representation. The used neural network is a graph convolutional network. This kind of network uses the information of adjacent nodes in order to make predictions in the graph. The results presented in this paper shows that the proposed system is able to predict next states with a precision close to 100% for a known use case, using manually generated data for training, validation and test.

Keywords: Social robots \cdot Graph Neural Network \cdot Robotics Cognitive Architecture.

1 Introduction

Service robots are currently receiving a growing interest in several application domains, such as industry, hospitality, health and social care, or home assistance, among others [1]. In these domains, robots must be able to interact with the users using natural channels in order to be useful [3]. One topic that has been explored in the last years for improving this interaction, is self-adaptive robotics [7]. Robots able to adapt to their social context must first monitor their internal and external context information. Then, they use this information to adapt their configuration and course of action at run-time, to better fit their behavior with what the user expects from them. Self-adaptation mechanisms used by these robots can benefit from the previous executions of the robot, in order to improve the interaction with the user in terms of acceptance and usefulness. In this vein, neural networks can be used to learn from previous experiences, allowing the robot not only to know the most probably next state at run-time, but also to simulate different scenarios before the robot is deployed on a real environment.

This work aims to infer the internal state of robots within the CORTEX cognitive architecture [2] which is currently used in various fields, including socially assistive robotics, driverless vehicles, intralogistics, and telepresence. The increasing demand for robots in these fields due to factors such as the aging population, the need to reduce the carbon footprint, the new demands in logistics, and the COVID-19 pandemic, is expected to lead to a significant growth in the application domains and perspectives of use of CORTEX in the coming years [1].

In CORTEX, the internal state of the robot is represented as a multi-labeled graph (the Deep State Representation, or DSR) that holds symbolic and geometric information within the same structure. The idea presented in this paper is to build a Graph Neural Network (GNN) capable of simulating and predicting the state of the DSR at any given time. The main reason to use a GNN to predict future states, instead of a different kind of network, is because the DSR is represented as a graph, and graphs are the inputs of the GNN. These predictions can help with simulating and developing future features for the robot. Moreover, predictive capabilities may allow the robot to adapt to its context. They also add an extra simulation layer to the robot, to check its behaviors before it is put into operation. This can help prevent future issues and speed up testing, as the state of the robot can be simulated without the robot being present, just by observing the internal state.

The built GNN collects and classifies information obtained from different simulations performed in the past. This neural network has been built from scratch. Each node and edge are the output of a classic neural network, which will in turn have the rest of the nodes and edges as input. This decision was made because no other network studied considers the values of the edges, although similar networks have been used in the past and have yielded good results [8], [4], [10].

The rest of the paper is organized as follows: Section 2 provides details about the CORTEX cognitive architecture. Section 3 describes our implementation, providing details about the most relevant components in our software architecture. Experimental results are presented in Section 4. Finally, conclusions and future work are drawn in Section 5.

2 CORTEX cognitive architecture

CORTEX [2] is a cognitive architecture based on three main points: modularity, internal model and graphic representation. CORTEX is also a framework designed to support early forms of intelligence in the real world and human-robot interaction, being able to select and decompose the robot capabilities. Those features has been translated to computational modules or agents that communicate with each other using a shared (and distributed) blackboard called Deep State Representation (DSR), which is able to store symbolic and geometric information



Fig. 1. An example of a CORTEX instance for a socially assistive robot.

as a multi-labelled graph. The DSR is a representation of the immediate context around the robot: the objects, the humans and the robot itself. All those entities are detected and transformed in different abstraction levels. Fig. 1 shows an example of a CORTEX instance employed for a certain socially assistive robot. As detailed above, software components connect via agents to a shared DSR. These components can be programmed using different languages and frameworks, and in fact they can be deployed in different hardware devices.

CORTEX agents are classified as perceptual, reactive, and deliberative agents. Perceptual agents are connected to sensors and update the DSR with the context information. Reactive agents aim to provide fast responses to changes in the DSR. Deliberative agents are responsible for making decisions, in order to achieve a given plan.

Figure 2 shows a simplified example of a DSR graph. In this example, the robot is in the same location than the person, and it is linked to other nodes that represent subsystems and actions, such as speaking, showing, battery, etc. Agents are reading the DSR to search for events/changes that trigger specific task-solving skills, and implement this mapping in their internal logic. Thus, they can annotate that currently the robot is not speaking or that the robot is charging its battery (using the *is_charging* edge between robot and battery). More details about the DSR can be found in [2].



Fig. 2. DSR initial state

3 Prediction agent

A cognitive architecture that counts with predictive mechanisms will be more agile in simulating and testing new algorithms and behaviours. This paper aims towards this feature. It proposes the inclusion in CORTEX of a deliberative agent able to predict the next states, that uses the DSR as input of a GNN (Graph Neural Network). GNNs are neural networks that work like an image recognition network, with the particularity that GNNs analyze graphs instead of images. Hence, the network will analyze each node, and the contiguous nodes linked to it by edges. These data will be used to predict next graph configurations and next robot action, in the same way that an image recognition network considers each pixel, and the contiguous pixels.

In order to create this agent we used a GNN with one convolution layer. It has one layer because we are considering, in this first implementation of the system, that there are no more than two edges between nodes (as shown in Fig. 2). It was assumed that our experiments do not build larger graphs, although the model used can be improved by adding more internal layers (no more than 2 or 3 [6]). The proposed GNN was built using the PyTorch geometric library, which provides the tools to write and train GNNs using different methods for deep learning in graphs¹. This network is used as a semi-supervised node with an adjacency matrix as input representing the graph [6]. The matrix has been built using the data of the graph and their vertex, creating a dataset of graphs, where each graph corresponds with a status of the DSR. The convolution layer (which is hidden), implements a typical convolution layer. It is connected to a linear output layer, which applies a linear transformation to the incoming data. Cross entropy has been used at the output, having as many outputs as possible states of the graph. Then, arguments of the maxima (argmax) have been used to find the biggest value of the output, corresponding with a label that represents the status of the graph. Those labels were previously converted to numbers, allowing them to be used as input/output of a neural network. Regarding the data acquisition section 4.2.

4 Experiments

In order to show how the GNN can predict the next states using the DSR, we used one of the use cases implemented by a Socially Assistive Robot (SAR) with the goal of proving how predictable the DSR is, and therefore, proving that robot actions can be predicted just using the DSR. The results generated from the GNN will be validated by comparing them with the outputs obtained from the use case execution.

4.1 Context and use cases

In this paper we used one of the use cases implemented by a Socially Assistive Robot (SAR) that integrates an instance of the CORTEX architecture, shown in Fig. 1. The basis of this robot is the CLARA robot [9], developed within the CLARC EU project ECHORD++ (FP7-ICT-601116). CLARA is currently deployed in a retirement home, where it performs different use cases. The experiments in this paper focus in one of these use cases, the *town crier* [5]. In this use case, the robot moves to a specific goal, and there it announces relevant information, such as the daily menu. The implementation of this use case requires different modules that read from sensors and update the DSR. For instance, the robot must be able to speak or navigate autonomously, avoiding obstacles and reaching different goals. All the context information and current state of the robot is available in the DSR. Thus, any agent running in the system can read and use it to perform specific tasks.

As it was detailed before, in the town-crier use case the robot must reach a specific goal and then reproduce a message to the residents. This message is also shown in the tactile screen located on the chest of the robot. The DSR stores all relevant information related to the use case, both symbolic an geometric.

¹ https://pytorch-geometric.readthedocs.io/en/latest/



Fig. 3. Initial state

One important node in the DSR is the *server* node, as it is the server agent the one in charge of requesting for new actions depending on the context. Robot actions themselves are annotated as nodes and linked to this *server* node, (e.g. see Fig. 12). Thus, when the robot wants to perform a specific action, a new action node is created and added to the DSR, and then a new *request* edge from the *server* node to the new action node is also created. An agent that implements the text-to-speech (TTS) algorithm will be aware of these changes and will change the DSR in order to start the use case reproducing the provided message on the robot screen.

The following items provide a deep description of how the DSR changes to make the robot speak a phrase (i.e. perform a *say* action):

- 1. Figure 11 shows the initial state of the DSR.
- 2. Once the message to say is shown in the robot screen, a new node called *say* appears (see Fig. 12). This node has an attribute called *text_to_say* that stores the message to be reproduced.
- 3. Then, as shown on Fig. 5, a new link between *robot* and *say* is created (the so called *is_performing*), and the robot starts speaking.
- 4. When the robot finishes speaking the edge between *robot* and *speaking* changes to *finish* (Fig. 6).
- 5. After that, *finish* is changed to *finish_with_planner_ack* as a finished confirmation from the robot is needed. This is a requirement from the planner agent that needs the end confirmation. This can be seen on Fig. 7.
- 6. Finally, the DSR goes back to the initial state (Fig. 7).

4.2 Data acquisition

The proposed approach is based on collecting and classifying all the information obtained from several executions of the use case performed on real scenarios. The data (log files) obtained from these executions were transformed into CSV



Fig. 4. Action node say appears.



Fig. 5. Edge between robot and say nodes appears.

files. Each CSV file contains information of a node, and each row represents a specific state in the DSR. The node information is stored in different columns:

- ID: An unique node identifier.
- action id: Identifier of the action performed by the robot.
- exists: It is a number that represents the existence of the node (1: exists, 0: does not exist).
- session id: Unique identifier to the node in the current session.
- node_from_x: Nodes connected to the current node.
- vertex_state_x: Label of the edges pointing to the node.
- name: Node name.
- last_action: It is the last action performed by the robot.
- action_group: It is the high level action the robot is performing.
- next_action: It is the expected next action.

Thus, a CSV file for each node is needed to represent a complete DSR graph. Figure 8 shows a CSV file example for a node called *say* and how it has evolved



Fig. 6. Finish edge appears between robot and speaking



Fig. 7. The label finish_with_planner_ack appears.

during the use case. The say node appears (exists column is updated to 1) in the DSR when the robot is performing the require_message action. At this moment of the use case, the say node is connected to a node with an id of 63 through the request edge label (see vertex_state). The next action to be executed by the robot will be playing_message, as the next_action column shows.

li	d	action_id	exists	session_id	node_from	vertex_stat	e name	last_action	action_group	next_action	node_from_	Ivertex_state_1
	100	init	0	1	-1	L	1 say		reproducing_message_from_ui	require_message_from_ui	-1	-1
	100	require_message_from_ui	1	1	63	request	say	init	reproducing_message_from_ui	playing_message_from_ui	-1	-1
	100	playing_message_from_ui	1	1	63	request	say	require_mes	reproducing_message_from_ui	finishing_message_from_ui	1	is_performing
	100	finishing_message_from_ui	1	. 1	63	request	say	playing_mes	reproducing_message_from_ui	finishing_message_from_ui_with_ack	1	is_performing
	100	finishing_message_from_ui_with_ack	1	. 1	63	request	say	finishing_me	reproducing_message_from_ui	message_from_ui_finished	1	is_performing
	100	message_from_ui_finished	0	1	-1		1 say	finishing_me	reproducing_message_from_ui	init	-1	-1

Fig. 8. Example of a CSV file.

As it was previously detailed, a node can be connected to more than one node. For instance, when the robot is performing the *playing_message* action

the say node is still connected to the node 63 with the request edge, but also to the node with id of 1 (see *node_from_1*) through the edge called *is_performing*.

4.3 Results

In our experiments, the GNN developed has been able to predict the next state of the DSR from the current state of the graph. The GNN returns the action name of the next state of the DSR, that corresponds with the name stored in the column *next_action* (check figure 8, column *next_action*). For instance, Fig. 9 shows a simplified representation of the current state of the graph (which is the initial state) used as input of the GNN, and Fig. 10 shows the next predicted state (the GNN output).



Fig. 9. Current state of the graph.



Fig. 10. Next state of the graph

As it is shown in the figures 9 and 10, a new edge has been created between nodes 5 and 6, which correspond with the nodes *server* and *say*. In the figures 11

and 12 it is shown how the graph has evolved in the current use case. Therefore, it can be observed that the predictions are correct as the graphs are similar. Also, the current graph and the next graph status are the same.

As Fig. 12 shows, the node 100 has been added, and also a link between the nodes identified as *server* and *say*. The comparison between figures 9 and 10, and figures 11 and 12, shows that the system has been able to successfully predict the next states for the DSR.



Fig. 11. Current state of the graph in the use case.



Fig. 12. The node say appears.

Following the use case, the next step will be the creation of a link between nodes *say* and *robot*, when the robot start speaking (Fig. 5). During the experiment, if the graph presented on the figure 12 (or figure 10) is used as input of the GNN, the output will address to the next status of the graph in the use case.

This output of the experiment can be shown in Fig. 13.

The next state of the graph, as the use case presents, will maintain the graph as it is, but will change the labels of the vertex. The vertex will remain on the DSR for two iterations, until the message ends. After it, the link between nodes *say* and *robot* and the link between *say* and *server* will disappear, bringing the graph to the initial state as Fig. 14 shows. This state corresponds with Fig. 11 in the use case.



Fig. 13. The node say links with the robot.



Fig. 14. The graph goes back to the initial status.

5 Conclusions and Future Work

In this paper, we performed the first steps towards using neural networks to infer the status of the DSR graph. We developed a GNN able to predict future status of the graph during a use case, obtaining an accuracy near to 100% predicting the next state of the graph. To reach those results, a procedure to acquire data from the robot has been implemented, creating the first available dataset of the DSR data. It opens a new world of possibilities in the robotic investigations, making it possible to predict the internal status of the robot, and its next actions.

Considering the obtained results in the performed tests, further work will focus on adding more values to the nodes and include their attributes in the prediction system. Hence, the DSR stores a lot more information than used in these tests. Moreover, the robot is currently performing several more complex use cases in the retirement home: data collected from the execution of these use cases in real environments will be included in the dataset.

All this information can be used to improve the graph predictions, aiming towards predicting how the attributes of each node change during the execution of different use cases. It is also interesting to include the labels of the links, and to extract the data directly from the robot logs in order to have a bigger dataset. Finally, using this bigger dataset, the possibility to automatically generate the graph, using generative AI, will be explored.

Acknowledgments. This work has been supported by grants PDC2022-133597-C42, TED2021-131739B-C21 and PID2022-137344OB-C32, funded by MCIN / AEI / 10.13039 / 501100011033 and by the European Union NextGenerationEU/PRTR (for the first two grants), and "ERDF A way of making Europe" (for the third grant).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

- Robotics 2020 multi-annual roadmap for robotics in europe. Tech. rep., SPARC: The Partnership for Robotics in Europe, euRobotics Aisbl: Brussels, Belgium (2015)
- Bustos, P., Manso, L., Bandera, A., Bandera, J., Varea, I.G., Martínez-Gómez, J.: The CORTEX Cognitive Robotics Architecture: use cases. Cognitive Systems Research (2019)
- Feil-Seifer, D., Mataric, M.: Defining socially assistive robotics. In: 2005 IEEE C9th Int. Conf. on Rehabilitation Robotics. pp. 465–468 (July 2005)
- Hamilton, W.L., Ying, R., Leskovec, J.: Inductive Representation Learning on Large Graphs. arXiv (2018)
- Iglesias, A., Viciana, R., Pérez-Lorenzo, J., Lan Hing Ting, K., Tudela, A., Marfil, R., Dueñas, A., Bandera, J.: Towards long term acceptance of socially assistive robots in retirement houses: use case definition. In: 2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC). pp. 134–139 (2020)
- Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. ICLR (2017)
- Romero-Garcés, A., Salles De Freitas, R., Marfil, R., Vicente-Chicote, C., Martínez, J., Inglés-Romero, J., Bandera, A.: QoS metrics-in-the-loop for endowing runtime self-adaptation to robotic software architectures. Multimed Tools Appl 81, 3603– 3628 (2022). https://doi.org/10.1007/s11042-021-11603-7
- Scarselli, F., Gori, M., Tsoi, A., M., H., Monfardini, G.: The graph neural network model. IEEE Transactions on Neural Networks 20(1), 61–80 (2009)
- Voilmy, D., Suarez, C., Romero-Garcés, A., Reuther, C., Pulido, J., Marfil, R., Manso, L., Lan Hing Ting, K., Iglesias, A., González, J., García, J., García Olaya, A., Fuentetaja, R., Fernández, F., Dueñas, A., Calderita, L., Bustos, P., Barile, T., Bandera, J.P., Bandera, A.: CLARC: A cognitive robot for helping geriatric doctors in real scenarios. In: ROBOT (1). Advances in Intelligent Systems and Computing, vol. 693, pp. 403–414. Springer (2017)
- Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)

On the use of ontologies as knowledge source for social robots the INSIGHT research project

Jesus Martínez-Gómez¹, Ismael García-Varea¹, M. Julia Flores¹, Luis de la Ossa¹, Javier Ballesteros-Jerez¹, Antonio Bandera², Rebeca Marfil², Pilar Bachiller³, and Pablo Bustos³

¹ Computing Systems Department, Universidad de Castilla-La Mancha, Spain {jesus.martinez,ismael.garcia,julia.flores,luis.delaossa, javier.ballesteros}@uclm.es ² Universidad de Málaga {ajbandera,rebeca}@uma.es ³ RoboLab, Universidad de Extremadura {pilarb,pbustos}@unex.es

Abstract. As in many other disciplines, social robotics has been influenced by the advent of new trends in massive data acquisition and processing. While there are several areas where algorithms directly trained from data are exceedingly useful, such as object detection or speech recognition, the core reasoning of a robot still requires knowledge provided by human supervisors, rather than being directly derived from data. In this work, we analyse the role of ontologies and causal reasoning for this purpose, and we review the necessary steps to adapt existing ontologies to a specific use case within the INSIGHT research project.

Keywords: Robotics · Ontologies · Causal Reasoning

1 Introduction

Understanding their environments as humans do has been a long-standing goal and aspiration for social robots. In this regard, robotics has evolved significantly from traditional reactive behaviours, where there was no need to understand the surrounding environment. Reactive paradigms were replaced by deliberative (or hybrid) ones, where robots identify and internalise the objects and agents around them, as well as the events occurring during their interactions with the environment. All these data are used, along with plans and strategies, to select the best robot actions. Deliberative reasoning commonly involves symbolic representations, but these representations do not always include additional knowledge about the inherent relationships among internal symbols. This issue negatively impacts two aspects. First, the robot may internalise symbols that are incoherent with both the current environment and symbols previously internalised. For instance, the symbol car or bus for a robot operating in a kitchen. Second, the lack of information between symbols may negatively affect the robot incremental learning from past experiences. The current work analyses the use of ontologies as a knowledge source for social robots, focusing on the scope of INSIGHT, a research project where causality and self-understanding are central themes. The project proposes a use case involving a robot named Shadow, which must follow its human supervisor. The robot goal is to distribute medicines in a hospital-like environment, and the use case presents an unexpected situation that cannot be explained by the robot current knowledge. This situation triggers the need for the robot to self-extend its knowledge, allowing it to either ground and internalise the unexpected event or discard it. The self-extension procedure we propose is illustrated in Fig. 1, and it includes different elements to be detailed in Section 5.



Fig. 1. Graphical abstract of the proposed solution to self-extend the robot knowledge

The rest of the article is organised as follows. Section 2 provides an overview of related works that help to understand the overall scope of the article. Ontologies and causal reasoning are covered in Sections 3 and 4, respectively. The use case proposed in INSIGHT is presented and analysed in Section 5, where the proposed solution is presented. Finally, Section 6 provides the conclusions and outlines future work.

2 Related Work

The underlying research scope of the INSIGHT research project is the application of causal reasoning for social robots, which can be considered one of the main trends in robotics [17]. This research direction diverges from the current focus on managing vast amounts of data to train algorithms that support decisionmaking, often without subsequent analysis of the learned implications. For instance, most current solutions in generative Artificial Intelligence (generative AI) rely on Large Language Models (LLMs). These models excel at emulating human speech (when trained on corpora generated from human activity), but they do not adhere to the formalism of formal grammar and are therefore prone to generating inconsistent sentences. Additionally, the unsupervised generation of these models increases their dependency on the source data. The logical alternative to language models, therefore, is the generation of grammars following formal procedures, a task traditionally undertaken by linguists and computer scientists.

In the field of social robotics, knowledge is often held by psychologists and sociologists, much like linguists hold expertise in language. This knowledge has been recorded and represented in various ways, but ontologies are perhaps the most popular method. The standardisation of syntaxes and languages, such as the Resource Description Framework (RDF) and the Web Ontology Language (OWL), has facilitated their widespread adoption. The use of ontologies in robotics is not new [8], and they have been applied to various traditional robotic problems, such as autonomous driving [5] and Simultaneous Localization and Mapping (SLAM)[10]. In related works, ontologies are typically used to conceptualise the domain in which the robot operates. In the case of SLAM, ontologies help to complement metric representations with semantic ones. The use of ontologies to manage temporal relations among events, such as ISRO [7], is particularly interesting, as they provide a dynamic semantic understanding of the robot behaviour and its surrounding agents and objects.

Causal reasoning relies on the existence of graph representations where arcs represent "cause" and "caused by" relationships. In [17], the author reviews and analyses robotics and causality with fascinating conclusions. One of the first points highlighted in this article is the low impact of causality in published research, which is noted to be contrary to other computer science disciplines. The hypothetical reasons for this, according to the author, are the gap between population-level and individual-level models, and the low-level sensors used for robots in contrast to the high-level concepts managed by causal reasoning. Another remarkable point from Hellström article is the review of how robots affect the world and their impact on causality. The combination of ontologies and causal reasoning has been explored in other works, such as the one presented in [2], where the resulting artefacts are causal Bayesian Networks [24], and ontologies are used for causal discovery. Biomedicine is a research area with a substantial presence of ontologies, and we can find services to support the hosting, search, and visualisation of biomedical ontologies, such as BioPortal, as well as works where the terms ontology and causality are managed together [22]. In robotics, the integration of causality and ontologies has not been as deeply explored as in other disciplines, but there are some related proposals. For instance, the authors in [9] develop an ontology-based Bayesian Network model for additive manufacturing. The interesting aspect of this article is that an existing ontology provides the prior knowledge to model causal connections.

3 Ontologies

3.1 Introduction

Ontologies are defined as formal specifications of shared conceptualisations [14], allowing the definition of terms and relationships within a topic area, thus comprising its vocabulary [21]. Other authors, such as [30], focus on the hierarchical nature of ontologies and define them as hierarchically structured sets of terms used to describe a given domain, which can serve as a skeletal foundation for a knowledge base. The potential use as a knowledge base is the primary reason for adopting ontologies in the INSIGHT research project. Ontologies are specified using formal languages like the Resource Description Framework (RDF) or the Web Ontology Language (OWL), and each ontology includes terms and relations between them within the scope of a specific domain. In addition to serving as a knowledge base and supporting domain-specific modelling and data organisation, ontologies have played a significant role in impacting semantic web technologies.

3.2 Components

An ontology \mathcal{O} consists of three main elements: concepts \mathcal{C} , relations \mathcal{R} , and instances \mathcal{I} . Concepts (also known as classes) represent the domain entities we want to specify, and they can have attributes. Instances (also known as individuals) are specific examples that belong to some of the concepts. Finally, relations define how concepts and individuals are related, and a given relation may also have attributes. In addition to these three elements, there are other features that depend on the language used for the ontology definition.

3.3 Classification

Ontologies can be classified based on their domain scope, as outlined in [12]. Following a top-down approach, we distinguish three different levels: foundational, domain, and application ontologies. Foundational ontologies represent the top level and, therefore, the most general category. Specifically, foundational ontologies describe basic, domain-independent relationships between general objects and concepts. Domain ontologies are restricted to a specific domain and perspective, making them more specific than foundational ones. Finally, application ontologies are used to conceptualise a domain from a single point of view, representing the most specialised level. Additionally, it is quite common to classify ontologies directly as top, medium, and low-level.

3.4 Examples

There are thousands of ontologies, especially in areas such as the semantic web and biomedicine. However, when focusing on robotic applications, this range is narrower. We will explore some examples in two of the previously mentioned categories: foundational and domain level. The two state-of-the-art foundational ontologies are formal proposals: the Basic Formal Ontology (BFO)[1] and the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE)[19]. From a social robotics perspective, the Socio-physical Model of Activities (SOMA)[3] is an ontology that could be considered foundational, or at least top-level. SOMA manages up to 538 concepts or classes with 262 attributes, and includes subclass relationships among them. For instance, "collision" is a subclass of "process", and "process" is also a subclass of "event" These subclass relations create a hierarchy that helps to better represent and organise concepts. DUL[26] is a top-level ontology based on DOLCE, encompassing generalist concepts and relations such as object, event, or transition. If we move on level down to domain ontologies in robotics, it is quite common to find proposals based on, or adapted from other ontologies. For instance, Perception and Manipulation knowledge framework (PMK) [11] proposes an ontology for perception and manipulation, which combines elements from top-level ontologies and concept instantiation. CARESSES [23] project uses a domain ontology that adapts ECHONET [25].

OCRA An ontology of special interest for the INSIGHT research project is OCRA [4]. This ontology is developed using a procedure guided by Competency Questions (CQs), with the objective of properly conceptualising collaboration processes. It includes concepts organised hierarchically, featuring terms such as "Object", "Agent", and "Social Agent", which are suitable for instantiating human and robot agents involved in collaborative tasks. Other relevant terms include "Event", "Collaboration", and "Process". The following sections of the article will provide additional information on how OCRA was generated and how this procedure can serve as a basis for ontology generation within INSIGHT.

3.5 Tools and Frameworks

Visualization Once an ontology has been selected for use or during an evaluation stage, developers need tools to perform various tasks, such as visualisation, editing, querying, and coherence checking. Visualisation is particularly important as it allows for a quick overall understanding of an ontology. One of the best tools for visualisation is WebVOWL⁴, which provides dynamic visualisation with search capabilities, filters, and adjustable label widths. WebVOWL also allows ontologies to be loaded directly from URLs. The latest version (1.1.7) includes an experimental editing mode. Figure 2 shows a visualisation of the SOMA ontology after searching for the concept "event".

⁴ https://service.tib.eu/webvowl/



Fig. 2. WebVOWL Visualization tool over the SOMA ontology

Edition and Querying Protégé⁵ is a popular solution for ontology management, and it includes features for visualisation, edition, instantiation, and inference mechanisms [20]. After loading an ontology, users are provided with multiple tabs to visualise and analyse the concepts/classes and their relationships, or to explore the individuals that have been instantiated. Fig. 3 illustrates these two functionalities of Protégé using OCRA as the ontology. In this case, the left diagram shows the class analysis centred on "event", and the right diagram explores the relations of the individual "kinova_robot", an instantiation of the class "Social Agent". Protégé also allows querying over the ontology classes and individuals. The querying process is very flexible, and we can include conditions like "has a relationship on", "type is one of", "is a subclass of" or "is an instance of", and combine them in the way we prefer.

Fig. 4 demonstrates how to use Protégé to query an ontology with individual instantiation. In this case, we use OCRA as the ontology, which conceptualises terms related to collaborations between humans and robots following a plan. The query allows us to retrieve all agents participating in a relationship with any goal, thereby identifying active agents involved in a collaboration.

Frameworks and Libraries Despite the value of tools like WebVOWL and Protégé, their use is primarily limited to the design and validation of ontologies. The application of ontologies in robotic developments necessitates the use of tools, such as coding libraries, that can be integrated with the software responsible for controlling the robot behaviour. This introduces considerations of memory and computational efficiency, as the robot must perform a large set of tasks in real-time, such as localisation, navigation, sensing, and interaction. Therefore, libraries for ontology management should offer facilities for efficient and lightweight operation.

⁵ https://protege.stanford.edu/



Fig. 3. Protégé tool over the OCRA ontology

Query		
Entities that match all	of the following:	
Is an instance of	 Agent 	
	Direct	
Has a relationship on	 isParticipantIn 	1
	that has a value that \sim	
	Has a relationship on	v nasGoal the second
		that has any value ~
0		
Execute		
2 results		
HumanOperator		
Kinova_robot		

Fig. 4. Query process using Protégé and the OCRA Ontology

The minimal functionalities required by libraries used in robotic developments include ontology loading, individual instantiation, and reasoning. Ontology loading can be performed during the robot warm-up phase, but individual instantiation and reasoning must occur concurrently with many other tasks. The primary objective of individual instantiation is to internalise the data collected by the robot, ensuring that symbols are complemented with semantic properties and relationships. Reasoning over the ontology is crucial during deliberative planning, involving queries whose answers may determine the robot subsequent actions.

One of the main libraries for ontology-oriented programming is OwlReady [18], which facilitates the execution of various actions on ontologies. Reasoning in OwlReady (version 2⁶) is managed by two popular projects: Hermit [13] and Pellet [29]. Reasoning enables the identification of relationships across concepts and includes ontology consistency checks that are crucial during individual instantiation. DeepOnto [15] is another library offering multiple ontology functionalities. It leverages deep learning and provides novel features such as ontology matching and verbalization. Specifically, deep learning is integrated by employing language models from the OntoLAMA dataset [16].

4 Causal Reasoning

Causality is one of the keystones of the INSIGHT research project, and it is inherently related to ontologies. In [27], we find an up-to-date article that analyses and reviews the representation of causality within ontologies. The work concludes that causality has been represented in very heterogeneous ways in ontologies, but it identifies a set of relationship types that could be associated with causality: "cause", "is responsible for", "associated with", "induces", "is triggered by", "influences", "leads to", "has etiology", "may cause", "can cause", and "may effect".

Causal reasoning presents a differentiated paradigm with a four-step analysis ⁷: a) model and assumptions, b) identify, c) estimate, and d) refute. Modelling and assumption involve generating a model of causal assumptions, guided by expert knowledge rather than learning from data. This is where ontologies can be particularly effective, as they already represent expert knowledge in specific domains. The resulting model should be an acyclic graph, with causal graphs, causal Bayesian Networks, or Potential Outcomes (PO) frameworks as the main options. Steps b) to d) allow the robot to identify the effect of its actions even before they are taken, thereby improving the robot behaviour. Identification and estimation are the steps in which the model and observed data are analysed, involving terms like interventions (actions) and observation (sensing), which must be carefully distinguished. Interventions are managed by the special "do" operator in causality, which generates a new representation/model where parent relations are removed from the affected concept. Finally, the refutation step checks and validates the consistency of the model with the data.

Frameworks and Libraries PyWhy⁸ is an outstanding framework for causal machine learning, encompassing specific solutions for inference (DoWhy) and knowledge discovery (CausalLearn). Specifically, DoWhy [28] is a library that supports various topics, including causal structure learning, interventions, counterfactuals, and root cause analysis. Another library capturing the interest of the research community is CausalNex⁹. This library manages causal reasoning using

⁶ https://pypi.org/project/owlready2/

⁷ https://causalinference.gitlab.io/book/

⁸ https://www.pywhy.org/

⁹ https://causalnex.readthedocs.io/

Bayesian Networks, which distinguishes it from its competitors. Additionally, CausalNex simplifies the generation of graph structures in a two-step procedure: a) allowing the user to manually incorporate edges representing domain expertise, and b) leveraging the data using structure learning algorithms.

5 Use case: the INSIGHT project

5.1 Introduction

INSIGHT is an ambitious research project aiming to: a) equip robots with the ability to navigate their surroundings, b) comprehend them by verifying the coherence of their internal models, and c) develop new causal explanations in response to detected inconsistencies. This work is focused on objectives b) and c), and assumes a social robot with navigation and interaction capabilities. The project relies on the existence of a robot named Shadow running an instance of the CORTEX architecture [6], both shown in Fig. 5. This architecture includes several memories (semantic, working, episodic) to be expanded, and an internal simulator under development/integration used, among other objectives, to support causal reasoning.



Fig. 5. Key elements in INSIGHT: the robot shadow (left) and the CORTEX cognitive architecture $% \mathcal{A}$

The primary objective of the working memory in CORTEX is to maintain an up-to-date representation of the robot and its surroundings, using a graph representation with nodes, arcs, and properties. Currently, the evolution of the working memory is not restricted by coherence checks, allowing new edges and nodes to be freely added or removed. Within the scope of INSIGHT, the role of the semantic memory is to provide additional knowledge about the objects in the working memory and their relationships. Therefore, the generation of the semantic memory is proposed to follow the same procedure as that for designing an application-level ontology. To achieve this, we initially need to determine (and restrict) the domain of INSIGHT. This has been done by defining a use case where the robot must accompany a human supervisor during a medicine delivery task. The robot will operate in a controlled indoor environment with human agents and a set of expected objects, such as chairs, tables, and medicines, commonly found in hospital-like buildings.

Any proposed modification to the working memory will be triggered by the robot internal agents (e.g., a chair detector fed with data from an RGB-D camera) and must be validated by the semantic memory. This validation can be viewed as an ontology process where certain individuals are instantiated while ensuring that the ontology remains coherent. A working memory modification may be rejected if: a) the semantic memory lacks the necessary knowledge, or b) the modification is semantically meaningless. INSIGHT addresses the first point by proposing evolutions of the semantic memory, ensuring that the new version remains coherent with the proposed modification.

The use case proposed in INSIGHT includes a scenario where the lack of knowledge in the semantic memory is explicitly addressed. Specifically, it includes a test case where the robot hits a bump and loses the medicine. This translates into an episode where the robot camera stops registering the medicine, prompting a modification of the working memory to remove the medicine. The most reasonable explanation for this event (medicine removed from the robot) is an action by the human supervisor. However, an interaction (e.g., conversation) with the supervisor will rule out this explanation. At this point, the semantic memory is evaluated to ensure that an evolution would be consistent with the removal of the medicine (this is where episodic memory and the simulator also come into play). In this test case, a plausible explanation for the unexpected event may involve concepts such as robot movement, robot collision with a bump, robot tilt, and medicine drop.

5.2 Solution Proposal

The solution to be explored in INSIGHT begins with an analysis of existing ontologies to design a domain-level ontology that encompasses all objects and concepts involved in the use case. This process may involve selecting and combining multiple ontologies, using appropriate tools as previously discussed. Once generated, the ontology can be pruned to remove concepts irrelevant to the robotic domain. This refined ontology is referred to as TMO in INSIGHT.

A second step in the solution involves selecting elements from TMO to generate an application-level ontology known as MLO. During this process, special attention will be given to relations that can be classified as causal, based on the list presented in [27] and previously enumerated. The resulting artifact, MLO, must be an ontology suitable for causal reasoning, ensuring that before we instantiate or remove a given individual, we can check the coherency of the resulting ontology. Additionally, causal reasoning will be employed to query the ontology and perform root cause analysis, aiming to identify potential causes for a given fact. For instance, the human supervisor could be proposed as a hypothetical cause for the medicine removal, even if this will ultimately be discarded through a conversation.

The third element of the proposed solution is the evolution of MLO to autonomously expand the robot knowledge. This evolution will be triggered when none of the hypothetical causes for a given event (e.g., the medicine falling) can be validated. The evolution of MLO will be managed as the search for MLO+ (an enhancement of MLO) by incorporating nodes and arcs from TMO. This search should be guided to include concepts and relations that could assist in the root cause analysis of the unexpected event, while also considering the efficiency of future reasoning processes. The validation of each MLO+ proposal will involve the simulator and the episodic memory to ensure that MLO+ is valid not only for the recent event but also for the robot past experiences.

6 Conclusions and Future Work

In this theoretical work, we have analyzed the use of ontologies as knowledge sources in robotics and outlined the objectives and challenges of the INSIGHT research project. We have reviewed works where ontologies and causal reasoning are applied, and discussed existing tools for managing solutions based on these technologies. The main conclusion we can draw from the literature is that, although ontologies are extensively used in robotics, they primarily support knowledge representation rather than playing a significant role in novel knowledge discovery. Furthermore, as noted by other authors, causality remains a paradigm with low impact in robotics literature, presenting substantial opportunities for exploration. As future work, our plan is to advance the proposed solution to generate a semantic memory suitable for integration into the CORTEX cognitive architecture.

Acknowledgments

This work has been funded by the project SBPLY/21/180225/000062 funded by the Government of Castilla-La Mancha and "ERDF A way of making Europe". This work has also been supported by Universidad de Castilla-La Mancha and "ERDF A way of making Europe" under project 2023-GRIN-34437, and grants PID2022-137344OB-C33, PID2022-137344OB-C31 and PID2022-137344OB-C32 funded by MICIU/AEI/10.13039/501100011033 and by "ERDF/EU".

References

 Arp, R., Smith, B., Spear, A.D.: Building ontologies with basic formal ontology. Mit Press (2015)

- Ben Messaoud, M., Leray, P., Ben Amor, N.: Integrating ontological knowledge for iterative causal discovery and visualization. In: European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty. pp. 168–179. Springer (2009)
- Beßler, D., Porzel, R., Pomarlan, M., Vyas, A., Höffner, S., Beetz, M., Malaka, R., Bateman, J.: Foundations of the socio-physical model of activities (soma) for autonomous robotic agents 1. In: Formal ontology in information systems. pp. 159–174. IOS Press (2021)
- Bezerra, C., Freitas, F., Santana, F.: Evaluating ontologies with competency questions. In: 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT). vol. 3, pp. 284–285. IEEE (2013)
- Brunner, S., Kucera, M., Waas, T.: Ontologies used in robotics: A survey with an outlook for automated driving. In: 2017 IEEE International Conference on Vehicular Electronics and Safety (ICVES). pp. 81–84. IEEE (2017)
- Bustos García, P., Manso Argüelles, L., Bandera, A., Bandera, J., García-Varea, I., Martínez-Gómez, J.: The cortex cognitive robotics architecture: Use cases. Cognitive Systems Research 55, 107 - 123 (2019). https://doi.org/10.1016/j.cogsys.2019.01.003, http://www.sciencedirect. com/science/article/pii/S1389041717300347
- Chang, D.S., Cho, G.H., Choi, Y.S.: Ontology-based knowledge model for humanrobot interactive services. In: Proceedings of the 35th Annual ACM Symposium on Applied Computing. pp. 2029–2038 (2020)
- Chella, A., Cossentino, M., Pirrone, R., Ruisi, A.: Modeling ontologies for robotic environments. In: Proceedings of the 14th international conference on Software engineering and knowledge engineering. pp. 77–80 (2002)
- Chen, R., Lu, Y., Witherell, P., Simpson, T.W., Kumara, S., Yang, H.: Ontologydriven learning of bayesian network for causal inference and quality assurance in additive manufacturing. IEEE Robotics and Automation Letters 6(3), 6032–6038 (2021)
- Cornejo-Lupa, M.A., Ticona-Herrera, R.P., Cardinale, Y., Barrios-Aranibar, D.: A survey of ontologies for simultaneous localization and mapping in mobile robots. ACM Computing Surveys (CSUR) 53(5), 1–26 (2020)
- Diab, M., Akbari, A., Ud Din, M., Rosell, J.: Pmk—a knowledge processing framework for autonomous robotics perception and manipulation. Sensors 19(5), 1166 (2019)
- Falquet, G., Métral, C., Teller, J., Tweed, C., Roussey, C., Pinet, F., Kang, M.A., Corcho, O.: An introduction to ontologies and ontology engineering. Ontologies in Urban development projects pp. 9–38 (2011)
- Glimm, B., Horrocks, I., Motik, B., Stoilos, G., Wang, Z.: Hermit: an owl 2 reasoner. Journal of automated reasoning 53, 245–269 (2014)
- Gruber, T., Translation, A.: Approach to portable ontology specifications. Knowledge Acquisition 5(2), 199–220 (1993)
- He, Y., Chen, J., Dong, H., Horrocks, I., Allocca, C., Kim, T., Sapkota, B.: Deeponto: A python package for ontology engineering with deep learning. arXiv preprint arXiv:2307.03067 (2023)
- He, Y., Chen, J., Jimenez-Ruiz, E., Dong, H., Horrocks, I.: Language model analysis for ontology subsumption inference. In: Findings of the Association for Computational Linguistics: ACL 2023. pp. 3439–3453 (2023)
- Hellström, T.: The relevance of causation in robotics: A review, categorization, and analysis. Paladyn, Journal of Behavioral Robotics 12(1), 238–255 (2021)

- Lamy, J.B.: Owlready: Ontology-oriented programming in python with automatic classification and high level constructs for biomedical ontologies. Artificial intelligence in medicine 80, 11–28 (2017)
- Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A., Schneider, L.: The wonderweb library of foundational ontologies. WonderWeb deliverable D 17, 2002 (2002)
- Musen, M.A.: The protégé project: a look back and a look forward. AI matters 1(4), 4–12 (2015)
- Neches, R., Fikes, R.E., Finin, T., Gruber, T., Patil, R., Senator, T., Swartout, W.R.: Enabling technology for knowledge sharing. AI magazine 12(3), 36–36 (1991)
- Noy, N.F., Shah, N.H., Whetzel, P.L., Dai, B., Dorf, M., Griffith, N.B., Jonquet, C., Rubin, D.L., Storey, M.A., Chute, C.G., et al.: Bioportal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Research 37, 170–173 (2009)
- 23. Papadopoulos, C., Hill, T., Battistuzzi, L., Castro, N., Nigath, A., Randhawa, G., Merton, L., Kanoria, S., Kamide, H., Chong, N.Y., et al.: The caresses study protocol: testing and evaluating culturally competent socially assistive robots among older adults residing in long term care homes through a controlled experimental trial. Archives of Public Health **78**, 1–10 (2020)
- Pearl, J.: From bayesian networks to causal networks. In: Mathematical models for handling partial knowledge in artificial intelligence, pp. 157–182. Springer (1995)
- Pham, V.C., Lim, Y., Sgorbissa, A., Tan, Y.: An ontology-driven echonet lite adaptation layer for smart homes. Journal of Information Processing 27, 360–368 (2019)
- Presutti, V., Gangemi, A.: Dolce+ d&s ultralite and its main ontology design patterns. In: Ontology Engineering with Ontology Design Patterns, pp. 81–103. IOS Press (2016)
- Sawesi, S., Rashrash, M., Dammann, O.: The representation of causality and causation with ontologies: A systematic literature review. Online Journal of Public Health Informatics 14(1) (2022)
- Sharma, A., Kiciman, E.: Dowhy: An end-to-end library for causal inference. arXiv preprint arXiv:2011.04216 (2020)
- Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner. Journal of Web Semantics 5(2), 51–53 (2007)
- Swartout, B., Patil, R., Knight, K., Russ, T.: Toward distributed use of largescale ontologies. In: Proc. of the Tenth Workshop on Knowledge Acquisition for Knowledge-Based Systems. vol. 138, p. 25 (1996)

Enhanced Robot Navigation in a Cyber-Physical System-of-Systems Architecture

Alberto Tudela, José Galeas, Óscar Pons, Juan Pedro Bandera, and Antonio Bandera

Avispa Group, ETSI Telecomunicación, Universidad de Málaga, Málaga-Spain {ajtudela, jgaleas,opfernandez,jpbandera,ajbandera}@uma.es

Abstract. The idea of introducing a robot into an Ambient Assisted Living (AAL) environment to provide additional services to those provided by the environment itself has been exploited in numerous projects. Moreover, new opportunities can arise from this symbiosis, which usually require both systems to share the knowledge (and not just the data) they capture from the context. Thus, using knowledge elaborated from the raw data captured by the sensors deployed in the environment, the robot can know where the person is, whether he/she should do some physical exercise, but also whether he/she should move a chair away to allow the robot to successfully complete a task. This paper describes how to extend the CORTEX software architecture to handle and merge, partially and task-specifically, the state representations built by the robot and the AAL system. In particular, we show how this symbiotic framework allows the robot to trace routes on a semantically annotated metric map updated with information obtained from sensors located in an AAL environment.

Keywords: Cyber-physical system-of-systems (CPSoS) \cdot Semantically-annotated metric map \cdot Robot navigation.

1 Introduction

A system-of-systems (SoS) is defined as a set of autonomous and independent systems (so-called constituent systems (CS)), providing a certain service [1]. Building a system-of-systems has the advantage that, in addition to the services that each system provides, new services emerge, which are based on the collaboration between the different systems deployed [2].

Basically, a cyber-physical system (CPS) is a physical system controlled or monitored by computer-based algorithms. The physical deployment will typically have sensors and actuators, distributed in a given environment. The computational deployment will also typically be distributed, with processing nodes connected to a network, which will manage the exchange of information. Decisionmaking will thus be supported by an internalisation of information captured from the real environment. Continuing with the definition used at the beginning of this Section, a cyber-physical system-of-systems (CPSoS) is an integration of independent CPSs, capable of providing services that can go beyond those provided by these same CPSs acting in isolation.

In this work, we make use of a CPSoS based on two independent CPSs: an AAL environment, equipped with sensors and computing elements, capable of monitoring the condition of the person living in a small apartment, and an assistive robot, capable of interacting with the person and assisting him/her in certain daily tasks. Both systems can be deployed independently and provide the services offered autonomously. In our case, both CPSs are designed to meet certain objectives, and will be centrally controlled. This type of CPSoS is known as directed [7], and in it, although the component systems may maintain the ability to work independently, the mode of work is subordinate to the overall purpose of the SoS. The main purpose of this article is not, however, to describe or present this CPSoS. Our focus is on presenting a framework that enables the integration of CSs at the knowledge level. This framework is an extension of the CORTEX architecture for cognitive robotics [14, 15], and is based on the existence of different representations of the state, built and maintained within each CS, and on the existence of a knowledge flow, which moves between them in order for the CPSoS to achieve the global objectives for which it was designed. To assess the validity of the proposal, a simple demonstrator has been designed, in which the metric map used by a mobile robot to trace routes through an indoor environment is augmented by the knowledge of the environment that the AAL system captures from it. This semantically annotated map allows the robot to know in advance, for example, that a door is closed or open, or that passage through a room is hindered by the arrangement of chairs. This allows the robot to ask a person to open a certain door or to move a chair aside, so that it can carry out the required action.

The rest of the paper is organised as follows: Section 2 discusses previous work in which either the autonomous robot or the AAL environment has been defined as CPSs. Previous work in which CORTEX has been used in assistive environments is also presented. Section 3 describes the proposed new framework for representing knowledge in which the robot and the AAL system remain independent elements, sharing a continuous flow of information. The experimental setting and the results obtained are presented in Section 4. Finally, Section 5 presents the conclusions and future work.

2 Related work

A CPS basically consists of highly integrated physical, control, communication and computational elements capable of managing a flow of information moving between the real physical world and the so-called cyberspace [8]. The most common scenario for deploying CPS is the so-called Industry 4.0. In order to guide the implementation of a CPS in factories, several architectures have been proposed. These architectures define the structure and methodology of CPS as guidelines. The 5-level CPS structure (the 5C architecture [8]) is probably the most popular implementation architecture [9]. The tiers in the 5C architecture
are: smart connection level, data-to-information conversion level, cyber level, cognitive level, and configuration level. At the first level, the sensors for data collection are chosen. The second level is the responsible of extracting relevant information from the raw data. The information sharing is the main topic at Cyber level. The generated thorough knowledge of the monitored system allows the Cognitive level to make decisions. At the Configuration level, the corrective and preventive decisions made in Cognition level are applied to the system. It is considered as the feedback from cyberspace to physical space.

In the context of Industry 4.0, it is common to integrate autonomous robots into the CPS built on the physical reality of the factory. It is therefore easier for the two topics to have converged within the framework of industrial robotics. For instance, from the 70 papers reviewed by Venancio-Teixeira et al [9], the most frequent task was pick-and-place (36 papers) followed by transport (24 papers) and assembling (13 papers). The most used robotic configuration was one arm (16 papers) or one automated guided vehicle (16 papers). CPS and Autonomous Robots are usually combined with other technologies (Internet of Things, Digital Twin, Big Data, etc.). This scenario, in which the robot is just another element in the CPS, is the most common [10]. However, robots sense, process, and react to information from the physical world. Usually, they are individual examples of complex CPS. The proposal by Lin et al. [3] describes a mobile autonomous robot that is framed in a computing network that can be described as a CPS. Following the principles of the 5C architecture, the architecture of this CPS is structured in five layers (Component, Intelligence, Cyber, Configuration and Deployment). In addition to the robot, the network includes edge computing nodes based on Raspberry Pi4 and Intel NCS2 for intelligent inference and data exchange. But we can also imagine the robot itself as a network, internally endowed with different edge computing nodes. Sensors allow the robot to capture data from the environment, which is processed by different computing nodes to obtain geometric and symbolic information. This knowledge will be the basis for decision-making and self-adaptation to changes in the environment. The robot is not the physical element of a CPS, but constitutes a complex CPS, capable of physical interaction with the environment, but also of abstracting from this reality and, for example, using a digital twin of itself to assess whether a mission can be carried out successfully [11]. When the robot and the environment maintain their own structures and tasks, the solution can be considered a CPSoS [13].

On the other hand, assistive environments for daily living (i.e., AAL) include the deployment of sensors and certain actuators in the home or residence where the person to be cared for lives, so that, with the help of the necessary computational management and decision-making mechanisms, the person can live a more autonomous life. Based on wireless network of sensors, integrated with software for home monitoring, Bocicor et al [12] describe an AAL system under the guidelines of a CPS. In the CPS proposal by De Venuto et al [4], the network includes wearable sensors. The proposal of Calderita et al [5] presents a CPS-AAL for caregiving centers. In addition to static sensors or actuators (environmental (to measure temperature, humidity, CO₂...), or person-centred (microphones, loudspeakers, cameras,...)), the system integrates a Social Assistance Robot (SAR). This proposal is built over the CORTEX software architecture [14, 15].

3 Shared state representation

Knowledge representation and reasoning is a symbolic branch of artificial intelligence that aims to design computer systems that reason about a machineinterpretable representation of the world, similar to human reasoning [17]. In this sense, knowledge is more than information, as it incorporates the agent's own experience or intuition. A knowledge-based system will have a computational model of the real world of interest [18], in which physical and virtual objects, events and actions, relationships between concepts, etc. are represented with symbols (and geometric information if necessary).

One of the main advantages of a CPSoS is that behaviours can emerge that individual CSs would not be able to address. This will require CSs to exchange knowledge with each other, not just information about their status [7]. In order to use these shared models, CSs must share the same formal definition of types, properties, and relationships between entities.

In our proposal we focus on run-time information exchange. Both the robot and the AAL will internally use a runtime model [16]. The structure of this model is presented in Section 3.1. The CORTEX software architecture, which is built to handle such a model, is also briefly presented. Then, Section 3.2 formalises our proposal.

3.1 CORTEX and the Deep State Representation (DSR)

Briefly, the CORTEX software architecture for cognitive robotics [14, 15] is based on the existence of a state representation or working memory, around which a series of software agents are organised to update or process it, using the already stored state information either to obtain higher level data or to make decisions. The architecture can therefore be classified alongside other proposals based on the blackboard concept. All information relevant to the system must be represented in the working memory, so that any agent has immediate access to this information and can react appropriately. The actions of the agents are synchronised by means of annotations, usually symbolic, in this same representation [16].

The core element in CORTEX is this central representation of the state, the so-called Deep State Representation (DSR). Although other, more longterm memories can be integrated into the architecture (map of the environment, database with up-to-date information on residents in a hospital environment, etc.), the DSR reflects the current, internalised situation of the environment. It has come to be seen as a digital twin of the outside world itself [5].

The DSR is organised as a directed graph, in which geometric or symbolic information is stored in both nodes and arcs [14]. The way in which a software agent accesses this information is relatively simple, allowing the design of



Fig. 1. The CORTEX instantiation in the proposal by Calderita et al [5]: AAL and robot share a common working memory

bridges with different operating systems or frameworks (ROS 2, Mira, Robo-Comp, MQTT, etc.). This facilitates the deployment of sensors and actuators and the integration of computational algorithms, as the agents that manage them can be implemented in virtually any software framework.

Figure 1 provides an overview of the instantiation of CORTEX in the CPS-AAL proposal [5]. The figure shows in the centre the DSR and, around this memory, services are distributed that are provided by the robot or by the AAL. It is important to note that this figure does not correspond exactly to how CORTEX is organised internally. In CORTEX, around the DSR, agents would be placed, which implement basic functions such as navigating, talking, listening, measuring temperature or humidity, capturing heart and respiratory rate, etc. In this figure, however, the emphasis is on how both elements (robot and AAL) share the same knowledge base and how this representation enables them to carry out different tasks. In the figure, the DSR shown is not complete. It does show a geometric reference (the one provided by the world node), from which the rooms hang. Each room has different sensors. At the moment, the person is in the bedroom, and the robot is in the kitchen-dining room. Geometric links are not shown, but they do exist and allow us to know the position of the robot



Fig. 2. AAL and robot share state representation, but maintaining their own working memories

or the person with respect to the reference provided by the world node. Other symbolic relations, or the attributes of the nodes, are not shown.

3.2 Sharing information between DSRs

As described in [16], the DSR can be considered as a graph-based runtime model, i.e. runtime knowledge. In our implementation there are two CSs: the robot and the AAL system. Both update and manage their own DSRs. Knowledge transfer takes place by sharing parts of these models. By doing so, each CSs can improve how it performs its tasks. For example, Figure 2 outlines this idea. The AAL system has sensors that can detect the person in any room of the environment. This information can be transferred to the robot, and when the robot needs to remind the patient to take a medicine, it can go directly to this person.

At design time, both DSRs are initialized with a distribution of the space. From here there is an agent developed in ROS 2, called *dsr_bridge*, in both CSs (robot and AAL). These agents allow both entities to share parts of its graph through ROS 2 topics, in this case, using FastDDS as a communication middleware. The knowledge transfer protocol works as follows:

- Both agents maintain its own DSR graph as a shared pointer that listen the changes taking place in the graph and also act as a ROS node. As soon as a modification of their DSR occurs, this modification is then published through the ROS *edges* and *nodes* topics. An attribute that we called "source" (robot or AAL) is added to the message, depending on the machine that publishes the change. The aim is not to 'auto-listen' and enter a publish/subscribe loop.

- When a publication has been made, the other corresponding agent (previously subscribed to the ROS topics) listens to this modification and updates its DSR with the knowledge received by the publication message (node name, type, attributes, link, etc...).
- When a modification message is received, all appropriate checks are made as to whether the node exists so as not to access nodes that are not in the DSR.

Finally, the messages published are filtered to prevent the other DSR from having irrelevant information from the other DSR. For example, the robot does not publish its transforms of the motors, cameras, base etc., but it does publish its battery level or its relative position with respect to the map.

4 Experimental results

The described framework has been evaluated in a simple use case: employing the information provided by the AAL system to augment the metric map used by the robot to navigate the environment.

Specifically, the environment is a small flat, with only three rooms: a kitchendining room, a bedroom, and a toilet. This small flat has been built in our research group's laboratory and we are currently sensorising it. Although it has different sensors for monitoring the state of health and position of the person, in the example we are going to show two sensors will be used: those that indicate that the doors of the flat are open or closed, and two 360-degree RGB cameras (360° Panoramic Indoor Fisheye Camera from Reolink), located above the kitchen-dining room and the bedroom.

To meet the objectives required of the CPSoS, the robot must navigate to different positions in the flat. In a normal case, the robot could trace a route to these positions using the available metric map of the environment, and then solve the problems it encounters on the fly. The advantage of having an augmented map, on which not only the position of the person can be located, but also that of moving obstacles such as chairs or tables, or doors that are open or closed, is that the robot can know in real time whether there are actually available passage routes. In case it needs to ask the person to open a door that is closed, or to move out of the way of chairs that are blocking the route, it can proactively request this help as a preliminary step to navigate to the desired position.

Below we detail the setting in which the experiment is carried out and an example of how both CSs collaborate to provide the robot with an augmented metric map, with semantic and geometric information on the position of people or other objects present in the two monitored rooms of the flat.

4.1 Experimental setting

The robot used in our experiments is a Morphia from MetraLabs GmbH. The Morphia robot is equipped with a Metralabs TORY differential base, a range



Fig. 3. The Morphia robot navigating in a crowed environment

laser scanner SICK s300 and a circular safety bumper. It also has an Intel RealSense D435i RGB-D camera facing the ground used for navigation, three 2MP Valeo cameras and a Microsoft Azure Kinect RGBD camera used for perception; and a tablet and speakers for interaction. Figure 3 shows the Morphia robot navigating in a real, crowed environment. To handle all of these sensors and actuators, the robot is equipped with an Intel NUC i7 and a NVIDIA Jetson Orin AGX.

The instantations of the CORTEX architecture that control the robot and the AAL system are shown in Figure 4. The software architecture supporting the AAL system has an MQTT agent, through which data is received from FMCW (Frequency-Modulated Continuous-Wave) radar sensors, magnetic sensors and environmental sensors. The FMCW sensors are radar devices that radiates continuous transmission power changing the operating frequency. They are able to measure ranges comparable to the transmitted wavelength with a high accuracy. Using a 60GHz sensor, they are able to capture the heart and breathing rates of people lying in bed or sitting on a chair in the kitchen-dining room. Environmental sensors provide measurements of temperature, humidity and air quality. Magnetic sensors let you know whether the two flat doors (main entrance and toilet) are open or closed. The two 360 cameras provide information on the position of the person(s) in the flat and on other objects in the two main rooms. A MicroROS agent allows a panic button to be connected to the DSR. The time sequence of data captured from the environment is stored from the DSR in a long-term memory.

On the other hand, the CORTEX architecture in the robot has the agents that the robot needs to navigate the environment (motor controller or battery monitoring is handled by MIRA, a software provided by MetraLabs GmbH, while the navigation stack is implemented in ROS 2). The robot also has its own agent to detect people, communicate verbally with the person, plan its actions using Behaviour Trees, or self-adapt to changes in the context. A WebServer



Fig. 4. Software modules on both CSs

agent allows it to communicate with the Chest screen. Finally, both architectures communicate using the aforementioned ROS bridges.

4.2 Robot navigation

The robot navigates autonomously using an enhanced version of the ROS 2 navigation stack 'Nav2' [21] that is managed by a DSR agent. This stack uses a metric costmap of the environment, captured at design time, to plan the paths to the different goals to be reached by the robot. This map has two layers: the global costmap that comprises the whole environment and a local costmap of 5m around the robot. The global costmap only includes fixed elements of the environment as walls, worktops, wardrobes, etc. On the other hand, the local costmap includes both moving obstacles, like people or chairs, and static elements but at a limited space around the robot.

This map, which the robot could still use as is to navigate the environment, will be augmented with the geometric and semantic information that the AAL system provides to the robot about people and objects in the rooms, as well as the opening or closing of doors. For this purpose, the robot's DSR is updated with the symbolic information of the objects present in each room (chairs, tables, people) or the opening/closing of doors, as well as with the necessary geometric



Fig. 5. (Top) Rectified images captured by the two 360 cameras; and (Bottom) associated depth maps

information (i.e. the projection on its own metric map of the position of each detected person or object). This geometric information is placed on top of the global costmap as a filter layer and allows the robot to know if, for example, chairs are in the path towards the medicine dispenser in the kitchen. In order to know these projections on the floor of the objects detected by the cameras, we will use a depth estimation, obtained using the ROS 2 wrapper DepthAnything-ROS [19] for Depth Anything model [20]. Figure 5 shows two depth images associated with RGB images captured using the 360 cameras. RGB images have been rectified to reduce distortions. From these images, as shown in Figure 6, the AAL system can detect the various objects or persons present in the rooms.

Finally, Figure 7 shows the metric map augmented with the information of objects/people present in the two main rooms of the flat, and the status of the two doors. As mentioned above, this map lets the robot know that a certain object, or a closed door, makes it impossible to reach a certain goal. In Figure 7(top) the door is open. The robot knows the position of people, chairs and tables, and plans a path to the bedroom. When the robot starts to navigate, the main door of the house closes (Figure 7(bottom)). The AAL system immediately detects this situation, and this knowledge is transferred to the robot, which stops and should inform the person that it cannot continue its mission unless the main door of the flat is opened. It is important to note that the robot will have this information in real time, as well as the possible relationships that the AAL system detects between the objects and people present in the flat. If the mission is to locate the person to remind him/her to take a certain medicine, the geometric position of the person will be available in real time as long as he/she is in the two monitored rooms. The robot will also be able to know if the person



Fig. 6. Detection of persons and objects in the kitchen-dining room

has entered the bathroom and must wait at the bathroom door for the person to come out.

5 Conclusions and future work

This article describes the integration of a robot with an AAL system in which both systems continue to operate independently and autonomously and can maintain their own tasks. The integration is carried out at the level of knowledge exchange. This transfer allows both systems to have an augmented knowledge, in which the relevant part of the other is incorporated.

As a use case to initially validate this proposal, the robot has been provided with a metric map augmented with semantic but also geometric information about dynamic elements and people. This knowledge would allow the robot to request specific help from the person to open a door or move a chair aside. To implement these behaviours, a new CS should be added to the architecture: the person itself. Both the robot and the AAL system would share with the person the vocabulary of terms (chair, table, door, open/close, move...), although now the connection to the person's internal DSR would not be a ROS bridge but a human-machine communication interface.

It is not immediately clear whether it is a better solution to maintain a single DSR, which communicates with agents deployed on the robot and the AAL system, or to maintain two complete independent architectures. As an advantage, we could cite the capacity that the latter implementation offers to verify the correct functioning of each system, as they are fully functional independently, as well as the ease of traceability of the use cases that are designed (as they remain



Fig. 7. (Top) The robot plans a path towards the bedroom. There is no problem to navigate this route. (Bottom) When the robot is moving, the main door is closed. This event is immediately detected by the AAL and communicate to the robot. The robot must ask to the person in the home to open this door.

confined to the environment of each system). The problem arises in deciding which part of the DSR is shared with the other.

Future work will involve designing the control system of the SoS itself, which will enable it to achieve its own objectives, different from those of the CSs that make it up. In the example shown in this article, the objectives are those of the CSs themselves: in the case of the robot, simply to achieve certain goals, and in the case of the AAL system, to monitor the state and position of the person in the flat. It is also our aim to incorporate, as already mentioned, the person into the CPSoS as a component. There is previous work in this direction, the so-called human-in-the-loop CPS (HiLCPS) [6] or social cyber-physical systems (SCPS) [22], which we will have to study in depth before undertaking this step.

Acknowledgments. This work is granted by PDC2022-133597-C42, TED2021-131739B-C21 and PID2022-137344OB-C32, funded by MCIN/AEI/10.13039/501100011033 and

by the European Union NextGeneration EU/PRTR (for the first two grants), and "ERDF A way of making Europe" (for the third grant).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

- 1. Jamshidi, M.: Systems of Systems Engineering. Wiley, Hoboken (2009)
- Kopetz, H., Bondavalli, A., Brancati, F., Frömel, B., Höftberger, O., Iacob, S.: Emergence in Cyber-Physical Systems-of-Systems (CPSoSs). In: Bondavalli, A., Bouchenak, S., Kopetz, H. (eds) Cyber-Physical Systems of Systems. LNCS, vol 10099. Springer, Cham (2016) https://doi.org/10.1007/978-3-319-47590-5 3
- Lin, H. -Z., Chen, H. -H. Choophutthakan, K. and Li, C. -H. G.: Autonomous Mobile Robot as a Cyber-Physical System Featuring Networked Deep Learning and Control, 2022 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Sapporo, Japan, pp. 268-274, (2022) https://doi.org/10.1109/AIM52237.2022.9863365
- De Venuto, D., Annese, V. F., Sangiovanni-Vincentelli, A.L.: The ultimate IoT application: A cyber-physical system for ambient assisted living, 2016 IEEE International Symposium on Circuits and Systems (ISCAS), Montreal, QC, Canada, pp. 2042-2045, (2016) https://doi.org/10.1109/ISCAS.2016.7538979
- Calderita, L.V.; Vega, A., Barroso-Ramírez, S., Bustos, P., Núñez, P.: Designing a Cyber-Physical System for Ambient Assisted Living: A Use-Case Analysis for Social Robot Navigation in Caregiving Centers. Sensors 20, 4005 (2020) https://doi.org/10.3390/s20144005
- Gil, M., Albert, M., Fons, J., Pelechano, V.: Engineering human-in-the-loop interactions in cyber-physical systems. Information and Software Technology 126, 106349 (2020) https://doi.org/10.1016/j.infsof.2020.106349
- Sanz, R., Bermejo, J., Rodriguez, M., Aguado, E.: The role of knowledge in cyber-physical systems of systems. TASK Quarterly 25(3) 355-373 (2021) https://doi.org/10.34808/tq2021/25.3/e
- Lee, E.A. The Past, Present and Future of Cyber-Physical Systems: A Focus on Models. Sensors 2015, 15, 4837-4869. https://doi.org/10.3390/s150304837
- Venancio Teixeira, J., da Silva Hounsell, M., Wildgrube Bertol, D. How CPS and Autonomous Robots are Integrated to other I4.0 Technologies: a systematic literature review. Production & Manufacturing Research, 11(1) (2023) https://doi.org/10.1080/21693277.2023.2279715
- Nikolakis, N., Maratos, V., Makris, S. A cyber physical system (CPS) approach for safe human-robot collaboration in a shared workplace. Robotics and Computer-Integrated Manufacturing 56, 233-243 (2019) https://doi.org/10.1016/j.rcim.2018.10.003
- Mazumder, A., Sahed, M.F., Tasneem, Z., Das, P., Badal, F.R., Ali, M.F., Ahamed, M.H., Abhi, S.H., Sarker, S.K., Das, S.K., Hasan, M.M., Islam, M.M., Islam, M.R. Towards next generation digital twin in robotics: Trends, scopes, challenges, and future. Heliyon 9(2) (2023) https://doi.org/10.1016/j.heliyon.2023.e13359
- Bocicor, M.I., Cuesta Frau, D., Draghici, I.C., Goga, N., Molnar, A.J., Valor Pérez, R., Vasilateanu, A. Cyber-physical system for assisted living and home monitoring, 2017 13th IEEE International Conference on Intelligent Computer Com-

munication and Processing (ICCP), Cluj-Napoca, Romania, 2017, pp. 487-493, https://doi.org/10.1109/ICCP.2017.8117052

- Mitchell, D., Blanche, J., Zaki, O., Roe, J., Kong, L., Harper, S., Robu, V., Lim, T., Flynn, D. Symbiotic System of Systems Design for Safe and Resilient Autonomous Robotics in Offshore Wind Farms. IEEE Access 9, 141421-141452 (2021) https://doi.org/10.1109/ACCESS.2021.3117727
- Bustos, P., Manso, L.J., Bandera, A., Bandera, J.P., García-Varea, I., Martínez-Gómez, J. The CORTEX cognitive robotics architecture: Use cases. Cognitive Systems Research 55, 107-123 (2019), https://doi.org/10.1016/j.cogsys.2019.01.003
- Marfil, R., Romero-Garcés, A., Bandera, J.P., Manso, L.J., Calderita, L.V., Bustos, P., Bandera, A., Garcia-Polo, J., Fernández, F., Voilmy, D. Perceptions or Actions? Grounding How Agents Interact Within a Software Architecture for Cognitive Robotics. Cogn Comput 12, 479–497 (2020) https://doi.org/10.1007/s12559-019-09685-5
- Romero-Garcés, A., Hidalgo-Paniagua, A., González-García, M., Bandera, A. On Managing Knowledge for MAPE-K Loops in Self-Adaptive Robotics Using a Graph-Based Runtime Model. Appl. Sci. 12, 8583 (2022) https://doi.org/10.3390/app12178583
- Grimm, S. Knowledge representation and ontologies. En Scientific data mining and knowledge discovery: Principles and foundations. Berlin, Heidelberg: Springer Berlin Heidelberg, 111-137 (2009)
- Zaraté, P., Liu, S.: A new trend for knowledge-based decision support systems design. Int. J. Inform. Decis. Sci. 8(3), 305–324 (2016)
- 19. Satoshi Tanaka. DepthAnything-ROS is ROS 2 wrapper for Depth-Anything (2024)
- Yang, Lihe and Kang, Bingyi and Huang, Zilong and Xu, Xiaogang and Feng, Jiashi and Zhao, Hengshuang. Depth Anything: Unleashing the Power of Large-Scale Unlabeled Data (2024)
- Macenski, Steven and Martin, Francisco and White, Ruffin and Ginés Clavero, Jonatan. The Marathon 2: A Navigation System (2020)
- Calinescu, R., Cámara, J., Paterson, C.: Socio-cyber-physical systems: Models, opportunities, open challenges. In: 2019 IEEE/ACM 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SESCPS). pp. 2–6 (2019)

CORTEX as an Enabler of Multi-platform Integration in Internet of Robotics Things ecosystems

Alberto Tudela¹, José Galeas¹, Óscar Pons¹, Carlos Madrid¹, Sara Aguilera¹, Pablo Gómez¹, Juan Pedro Bandera¹, Antonio Bandera¹, and Pablo Bustos²

¹ Avispa Group, ETSI Telecomunicación, Universidad de Málaga, Málaga-Spain {ajtudela, jgaleas, opfernandez, jpbandera, ajbandera}@uma.es

² RoboLab, Escuela Politécnica, Universidad de Extremadura, Cáceres-Spain pbustos@unex.es

Abstract. The integration of robotic technology in an Internet of Things (IoT) environment is known as the Internet of Robotics Things (IoRT). The benefits that this integration can bring have been highlighted in recent publications. One of the application scenarios is the Ambient Assistant Living (AAL) environment. The environment is equipped with appropriate sensors to provide the robot with more detailed knowledge of the context, while the robot can move around the environment and interact with the people in it, capturing knowledge that can help to improve the AAL's behaviour. This paper describes the deployment of an IoRT ecosystem within a small apartment based on the CORTEX software architecture. The use of CORTEX has allowed us to integrate different frameworks (wireless network, micro-ROS, ROS 2, RoboComp, Mira) into this ecosystem relatively easily, connecting them all to the same knowledge representation. In addition to other use cases, the main goal of this deployment is to be able to monitor both the position and certain vital parameters of an elderly person living alone at home.

Keywords: Internet of Robotics Things (IoRT) \cdot Ambient Assistant Living \cdot Multi-platform integration

1 Introduction

The ageing of the population in developed and developing countries, together with the maturity of certain technologies, means that many institutions, both public and private, are seriously considering the design of care environments for the elderly with a significant degree of technological innovation. An Ambient Assisted Living (AAL) environment is an integration of independent assistive technologies, solutions and services. AAL (or simply assisted living) solutions can positively influence the health and quality of life of people, especially the elderly. Ambient Assisted Living (AAL) environments include the deployment of sensors and actuators in the home or residence where the person to be cared for lives, so that, with the help of the necessary computational management and decision-making mechanisms, he or she can lead a more autonomous life [1].

The AAL paradigm benefits from the growing impact of other technologies. In particular, the Internet of Things (IoT) aims to extend the internet to real-world objects, connecting smart and sensing devices into a global network infrastructure that includes physical and virtual objects. If we are able to use it properly, the IoT has the potential to increase the quality of life of inhabitants and users of intelligent AAL environments [2]. Another relevant technology that can be incorporated into the AAL environment is autonomous robotics. The combination of Robotics and Internet of Things (IoT) is known as the Internet of Robotics Things (IoRT). The presence of a robot in such ecosystem will mean increasing the still excessive price of home automation. However, the robot will be able to solve one of the problems that interacting with an intelligent environment can pose, for an elderly person, by providing a more intuitive and natural interface for interaction. The design of graphical interfaces, accessible via smartphones or digital tablets, will be a cheaper option. However, as several studies have shown [3,4], evaluation of an older person's reactions to recommendations provided by a robot or a tablet show that they perceive the robot as more usable and reliable than the tablet. The major advantages of the robot are that it has a physical embodiment and can operate with a certain degree of autonomy. Thus, it is able to add to the interaction with the person specific features that, obviously, the tablet or smartphone will not be able to do. In fact, it is becoming increasingly common for the AAL environment to be equipped with a robot that assumes, in this ecosystem, this role of interface with the person [5, 12, 8]. Moreover, the robot can benefit from the recent popularisation of artificial intelligence-based solutions to personalise its attention to the person and provide services that were unimaginable just a few years ago.

This article presents an IoRT proposal for AAL that integrates different sensors to track and monitor certain vital parameters of a person. The system incorporates a robot, also equipped in our case with numerous sensors, which allows it to navigate in the environment and interact with the person. The software architecture used is CORTEX, organised around a working memory that internalises knowledge of the current state of the environment in a directional graph (the Deep State Representation, DSR). This graph is a runtime model, which offers the important advantage of allowing the connection of software components implemented on different hardware platforms or operating systems. In addition, this architecture successfully addresses the challenge of integrating the various platforms present in an IoRT deployment. In particular, the robot has software components developed in RoboComp, ROS 2, and Mira. In the IoT environment, there are software components embedded on microcontrollers and single board microprocessors, using software environments such as micro-ROS (with FreeRTOS) and MQTT.

The rest of the article is organised as follows: Section 2 briefly describes different examples of robot deployment in IoRT systems for AAL. Section 3 describes the hardware components and software agents deployed in our implementation. Section 4 presents some example data captured by the ecosystem. Finally, Section 5 presents conclusions and future work.

2 Related work

2.1 The multi-platform challenge

AAL technologies encompass a wide variety of platforms, which are applied in multiple contexts. Focusing the application framework on elderly care, and taking into account the views of healthcare professionals, end-users and family members, it can be established that the key objectives of AAL are to increase the autonomy and safety of end-users, and to reduce the workload of caregivers and healthcare staff [6]. The key concept to achieve these goals is monitoring in the elderly person's environment. This monitoring enables the automatic and immediate detection of any emergency, as well as a quick check of the person's condition and location. The most immediate way to try to achieve this goal is to deploy sensors in the environment in which this person lives. Devices can be wired or wirelessly networked and, depending on the computational power required, will typically be single-board microprocessors or microcontrollers. The presence of the robot in the AAL ecosystem can help to carry out this monitoring, not only because it incorporates sensors that can capture vital parameters related to the person's state of health, but also because it can talk to the person, and thus obtain data that cannot be captured by sensors. But, of course, the robot provides its capacity as an interface for interaction [7], facilitating the elderly person's communication with family and friends, but also with external experts, caregivers or other residents [8]. By incorporating the robot, the ecosystem will usually include more frameworks and different platforms: computers, graphics processing units, multiprocessor system-on-chips... This will further complicate the software architecture, forcing it to be able to manage this complex multiplatform environment.

Finally, the robot can be used for therapeutic purposes, to manage rehabilitation through serious games, or to capture Comprehensive Geriatric Assessment (CGA) data [9, 1]. When analysing contributions on robotics deployed in AAL ecosystems, Abdi et al. [9] distinguish between service robots, which aim to assist users in their daily activities, and companion robots, such as the one described by Wada et al. [10], which aim at the psychological well-being of the people they accompany. Focusing on the first category, service robots, it can be concluded that they currently have two main drawbacks: (1) they usually offer very simple functionalities, mainly associated with supervision, with a gap between these functionalities and those required by end users, which would require the design of more complex but equally robust robot performance planning and control schemes; and (2) the deployment of these robots in a real intelligent environment requires the maintenance of a unique representation that brings benefits to both robot and environment. This is still in its early stages, with few projects showing evaluations obtained from actual long-term deployment [11]. To address these problems, the robot will typically receive computational support from an external server, so the ecosystem can incorporate new frameworks. The final result is that the IoRT environment deployed for AAL usually includes IoT and autonomous robots, but also smart connectivity, distributed and federated edge/cloud computing, and Artificial Intelligence (AI). In some cases, it can also consider Digital Twins (DT), Distributed Ledger Technologies (DLT), Virtual/Augmented Reality (VR/AR) and swarm technologies. It is clear that it has a strong multi-platform character, which can be a relevant challenge when a real deployment is addressed.

2.2 Expected services of IoRT in AAL

To enable robots deployed in the AAL to offer more complex services, the degree of integration with sensors deployed in the environment needs to be improved [13, 14]. In the GiraffPlus project [12], the environment where the person lives is equipped with a network of sensors, which are responsible for monitoring the person's condition. When certain states are detected in the person (nervousness, anxiety...), the robot serves as a tool to put the person in contact with family or friends. The main limitation of this robot is that it is teleoperated. In the MORPHIA project [8], the robot is also used for closing the communication gap between users and caregivers. Thus, elderly people can use the robot to communicate, via video or chat, with relatives, friends or caregivers. The robot can remind users to take medication, or to transport meals or personal items within the home (as the Turtlebot robot deployed in the RmR AAL ecosystem [15]). When an incoming call comes in, the robot can look for the user. Relatives living far away can use an intelligent remote control to keep an eye on things in the home or support them in certain activities via telepresence. Using static sensors and the ones endowed in the robot, the RiSH proposal [16] addresses human body activity recognition. Provided applications are human position tracking and human activity monitoring. The recognition of activities, normal or abnormal, is also the service provided by the AAL environment and the robot in [17]'s work. The proposed framework includes long-term memory models (LSTM) as well as reasoning based on Probabilistic Answer Set Programming (PASP). In all these cases, the AAL ecosystems employ the sensors deployed in the environment and in the robot for completing the ongoing task. In other proposals, the robot acts as a virtual therapist, which proposes the user physical or cognitive activities [18, 19]. Similarly, the robot can stimulate the user it accompanies to perform everyday activities [20]. Other functionalities are fall detection [21], meal assistance [22] or information and stimulation through messaging [23]. In the aforementioned ALMI project, the TIAGo robot uses both its speech interaction for voice instructions and its object manipulation capabilities to help a user with mild motor and cognitive impairments to prepare a meal. Speech interaction is also the main functionality deployed by the robot in the proposal by [24].

Calderita et al. [19] propose the use of CORTEX [25] to build the software architecture that controls the entire AAL ecosystem. In this case, the knowledge

that the architecture internalises from the environment is unique, and incorporates for example all sensor data, internal or external to the robot. Decisionmaking is carried out outside the robot. This article describes a very similar scenario, where CORTEX is adopted as the software architecture of the entire ecosystem.

IoRT instantiation using the CORTEX architecture 3

3.1CORTEX and the Deep State Representation (DSR)

CORTEX is a multi-agent architecture designed to facilitate the creation of information flows among different types of memories and modules. These flows are pumped by agents. Agents are represented as semicircular forms connected to the working memory \mathcal{W} , the so-called Deep State Representation (DSR). They can edit it to create and maintain an updated context accessible to the rest of the agents. W is a distributed graph structure implemented using conflictfree replicated data types (CRDT) [26]. Its nodes are elements of a predefined ontology, and the edges can encode geometric transformation or logic predicates. Both nodes and edges can store a list of attributes of a predefined type.

In this paper, we use a simplified version of the CORTEX robotics cognitive architecture [25]. The full version is shown in Figure 1. For our experiments, we use the working memory and a set of agents that pump information in and out of it, creating a stable representation for the robot's state and the monitoring devices placed in the environment.



Fig. 1. The CORTEX architecture

3.2 Multi-platform integration

The implementation of the working memory in the initial version of CORTEX was based on the concept of a real-time database [25]. In this implementation, one software module stores, receives and publishes changes to all agents in CORTEX. This version of CORTEX was successfully endowed into multiple robotic platforms. On the CLARA robot, all software components were implemented using RoboComp (https://robocomp.github.io/web/) except for the navigation and localisation stack which, provided by the robot base manufacturer (MetraLabs GmbH), used Mira (https://www.mira-project.org/joomla-mira/index.php/resources/repositories). A bridge was designed to connect the two frameworks, allowing the rest of the architecture to handle navigation-related commands. This bridge worked well, but we cannot say that Mira was fully integrated into CORTEX. The navigation or localisation data remained internal to the Mira part of the architecture, without being reflected in the working memory.

As aforementioned, in the latest version, the working memory is a distributed graph structure implemented using conflict-free replicated data types (CRDT) [27]. Significantly, the new implementation is not tied to any specific framework. Furthermore, being aware that the fundamental idea of CORTEX is that all relevant knowledge is available in real time to all agents in the architecture, the newly developed bridges adequately manage the updating of the working memory. In the AAL deployment that serves as the backbone of this article, the CORTEX architecture deployed includes bridges that connect the working memory with microcontrollers, single-board microprocessors, and the Morphia robot. Internally, the robot includes software agents developed in ROS 2, RoboComp, and Mira. Details on some of these bridges are provided below.

The MQTT agent The MQTT agent is in charge of interfacing between the DSR and sensors designed on microcontrollers or single board microprocessors, equipped with a WiFi transceiver. This agent will be in charge of receiving the information through the MQTT messaging protocol (https://mqtt.org/) and updating it in the DSR. The agent has been developed in C/C++ and basically consists of two distinct parts:

- Connection with the DSR: This agent connects to the DSR of the intelligent environment through a shared pointer. In addition, the agent responds to signals or callbacks that are executed when a modification occurs in the DSR
 - Creation of a node: this signal is triggered when a new node is inserted in the DSR.
 - Modification of an attribute of a node: this signal is triggered when an attribute of any node is modified.
 - Creation of an arc: this signal is triggered when an arc is created between two nodes in the DSR.
 - Deletion of a node: executed when a node is deleted from the DSR.
 - Deletion of an arc: executed when an arc is deleted from the DSR.

- MQTT Subscriber and Publisher: The PahoC++ library from the Eclipse Foundation has been used to implement this protocol³. Through this library, the connection to the Mosquitto broker is made as a client, subscribing to the determined topics, both to receive the desired information, and as a publisher to notify the external components of any relevant change in the DSR that may alter the operation of these components.

As mentioned above, the implementation of these agents aims for full integration in CORTEX. As an example of how the MQTT agent works, we can describe how a radar sensor, deployed under the bed and in charge of estimating heart and respiratory rates, is informed to start measuring. A zenithal camera is deployed in the bedroom, connected to the DSR by a ROS agent. When this camera detects that there is a person in bed and notes it in the DSR, the MQTT agent detects this situation and publishes the message 'OnSensor' through the MQTT Control topic, so that the microcontroller that manages the radar sensor activates it and starts taking measurements. Likewise, when the person gets out of bed, the agent in charge of managing the camera updates the DSR, and the agent publishes the message 'OffSensor', so that the measurements are finished, avoiding erroneous measurements that could cause confusion and false alarms.

The MicroROS agent Micro-ROS (https://micro.ros.org/) brings all major core concepts of ROS, such as nodes, publish/subscribe, client/service, node graph, lifecycle, etc., onto microcontrollers (MCU). The client API of micro-ROS (in the C programming language) is based on the standard ROS 2 Client Support Library (rcl) and a set of extensions and convenience functions (rclc).

The main targets of micro-ROS are mid-range 32-bits microcontroller families. Usually, the minimum requirements for running micro-ROS in an embedded platform are memory constraints. Specifically, in the deployment described in this paper, the ESP32-WROOM32 microcontroller was used (see Section 4.1, combining micro-ROS and FreeRTOS functionalities.

The MicroROS agent is provided by the microROS community. It runs in the Intel NUC that centralizes the management of the IoRT system.

ROS 2 and RoboComp agents Both inside the robot and in the IoT system, there are different agents implemented in different frameworks (in our case, in RoboComp and ROS 2). Depending on the specific task for which they are implemented, these agents update the DSR to include new nodes (e.g. the person node) or to incorporate more information in the form of attributes (e.g. the name of the person once identified). In addition, they notify the software components to which they are connected, when a node is created or an attribute is modified in a node or arc. This allows them to respond to this change in the DSR. When describing the MQTT agent, we have described an example of these situations, but this is the way it works in CORTEX.

³ https://eclipse.dev/paho/files/mqttdoc/MQTTAsync/html/index.html



Fig. 2. The Morphia robot navigating in the small apartment

4 Experimental results

4.1 Components of the AAL ecosystem

The Morphia robot The assistance robot used in our AAL is a Morphia from MetraLabs GmbH. Developed in the course of the joint research project MORPHIA (https://www.morphia-projekt.de/), it is the result of a collaboration among the Technical University of Ilmenau, CIBEK Technology und Trading GmbH, SIBIS Institut für Sozial- und Technikforschung GmbH, the University of Osnabrück, AWO Alten- Jugend- und Sozialhilfe (AJS) GmbH, and YOUSE GmbH.

Figure 2 provides some snapshots of the Morphia robot in the small apartment where the IoRT has been deployed. The Morphia robot is equipped with a Metralabs TORY differential base, a range laser scanner SICK s300 and a circular safety bumper. It also has an Intel RealSense D435i RGB-D camera facing the ground used for navigation, three 2MP Valeo cameras and a Microsoft Azure Kinect RGBD camera used for perception; and a tablet and speakers for interaction. To handle all of these sensors and actuators, the robot is equipped with an Intel NUC i7 and a NVIDIA Jetson Orin AGX.

The FMCW sensor for hearth/breathing rates estimation Two of the most important vital parameters of a person are their heart and respiratory rates. To measure them, two frequency modulated continuous wave radar (FMCW radar) sensors have been deployed in the IoRT system. This device is a special type of radar sensor that emits a continuous transmission signal like a simple continuous wave radar (CW radar). In contrast to this radar, an FMCW radar can change its operating frequency during the measurement, i.e. the transmitted



Fig. 3. Device designed for mounting the FMCW 60GHz sensor (MR60BHA1) from Seeed Studio and the ESP32C3 microcontroller: (Left) 3D model generated by Kicad 8.0; (Middle) Top layer showing the ESP32C3; and (Right) Bottom layer showing the MR60BHA1 sensor

signal is frequency modulated. These changes in frequency make further measurement possibilities technically possible through time-of-flight measurements.

CW radars have the disadvantage that they cannot measure distance due to the lack of time reference. In an FMCW radar, this time reference is generated by frequency modulation of the transmitted signal. With this method, a signal is emitted that periodically changes frequency. If an echo signal is received, this frequency change has a time delay Δt , like a pulse radar.

The main advantages of an FMCW radar are that it can measure very small target distances (the minimum measurement range is in the wavelength range), that it can measure distance and radial velocity simultaneously, and that the accuracy of the distance measurement is very high. In its 60GHz version, this device can measure the heart and respiratory rates of the person in front of the sensor, completely non-invasively, and without requiring the sensor to be pointed at a very specific area of the person.

Using Seeed Studio's MR60BHA1 as the sensor, and an ESP32C3 microcontroller as the control element, Figure 3 shows the design of a fully autonomous device to measure heart and respiratory rate and send this data via MQTT. The device can be placed under the bed or behind the back of a sofa, and measure these frequencies when the person lies or sits down.

The 360° zenithal cameras The tracking of the person(s) moving around the small flat is carried out using two 360° zenithal cameras located above the two main rooms of the apartment: the kitchen-dining room and the bedroom. Specifically, the cameras used are the Panoramic Indoor Fisheye Camera with 6MP SHD from ReoLink. Figure 4(left) shows what this camera looks like.

In our case, the cameras have been connected to a router, which allows them to be made available to the stack software in charge of processing them. This processing basically consists of rectifying the images, obtaining a depth estimation, and detecting the objects present in the image. The depth estimation is



Fig. 4. (Left) The 360° Panoramic Indoor Fisheye Camera with 6MP SHD from ReoLink; and (Right) a capture of the kitchen-dining room and the associated depth map.

carried out using using the ROS 2 wrapper DepthAnything-ROS [29] for Depth Anything model [30].

Environmental measures To measure the most relevant environmental parameters of the environment, a second board has been designed, also with an ESP32-C3 as a control and data sending element using MQTT, and with a BME680 sensor as a measuring element. This sensor provides data on temperature, humidity, atmospheric pressure and Volatile Organic Compounds (VOC).

4.2 Obtained results

Figure 5 shows a snapshot of the state of the DSR at a certain instant in time. Specifically, the captured situation shows that the robot is standing still, and interacting with Oscar. Both the robot and the person are in the bedroom, where a 360 zenith camera and an FMCW sensor are installed. In the other important room, the dining room, there are distance sensors, a 360° zenithal camera and an environmental sensor (marked as a temperature sensor). Each node of the network stores information captured from the sensors installed in the environment or onboard the robot. The temperature of the dining_room, for example, is 24 degrees, and Oscar's heart rate is 72 bpm.

While Oscar was lying in bed, the FMCW sensor captured his heart and respiratory rate data. Figure 6 shows some of the captured data. The heart rate and respiratory rate data are characteristic of a healthy person.

Figure 7 shows the detection of objects and persons using the camera above the bedroom. The system allows us to track the movement of the person through



Fig. 5. A snapshot of the DSR running within the IoRT ecosystem

almost the entire flat (the bathroom has been left out, although we can detect that the person has entered or left the bathroom).

5 Conclusions and future work

This article shows the deployment of an IoRT ecosystem for AAL. In terms of software, the CORTEX architecture has been used as a basis. The need to integrate different platforms has necessitated the design of different bridges, an issue that has been relatively easily solved thanks to the CORTEX design philosophy.

Our efforts are currently focused on completing the assembly of the small flat in which the devices discussed in this article have been deployed. In this process, new sensors and actuators will be added to the IoRT ecosystem, allowing the robot, for example, to command a motorised door to open. New sensors, such as the described FMCW radar, will also be added to the robot. Although we currently only use it to estimate heart or respiratory rates, there is work using it to identify people. This could be very interesting when the robot is deployed in environments where there are more residents, such as a nursing home. On the other hand, the deployment of such a complex ecosystem around a single working memory makes it difficult to validate, on certain occasions, the functioning of a specific part of it. Another option we are evaluating in parallel is to split the ecosystem into the two fundamental parts of the ecosystem (i.e. IoT and

Fig. 6. Hearth and breathing rates captured using the FMCW sensor

robot), each with its own working memory and associated tasks. Obviously, both components should be connected and share knowledge, but they would maintain an independence that, we understand, facilitates verification and maintenance.

Acknowledgments. This work is granted by PDC2022-133597-C42, TED2021-131739B-C21 and PID2022-137344OB-C32, funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGeneration EU/PRTR (for the first two grants), and "ERDF A way of making Europe" (for the third grant).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

- Blackman, S.; Matlo, C.; Bobrovitskiy, C.; Waldoch, A.; Fang, M.L.; Jackson, P.; Mihailidis, A.; Nygård, L.; Astell, A.; Sixsmith, A.: Ambient Assisted Living Technologies for Aging Well: A Scoping Review. Journal of Intelligent Systems, 25, 55–69 (2016) https://doi.org/10.1515/jisys-2014-0136
- Hail, M.A., Fischer, S. IoT for AAL: An Architecture via Information-Centric Networking. 2015 IEEE Globecom Workshops (GC Wkshps), San Diego, CA, USA, 2015, pp. 1-6, https://doi.org/10.1109/GLOCOMW.2015.7414020
- Hammer, S., Kirchner, K., André, E., Lugrin, B. Touch or Talk? Comparing Social Robots and Tablet PCs for an Elderly Assistant Recommender System. In Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction (HRI '17). Association for Computing Machinery, New York, NY, USA, 129–130 (2017) https://doi.org/10.1145/3029798.3038419
- Deublein, A., Lugrin, B. (Expressive) Social Robot or Tablet? On the Benefits of Embodiment and Non-verbal Expressivity of the Interface for a Smart Environment.



Fig. 7. Detection of objects and person/s

In: Gram-Hansen, S., Jonasen, T., Midden, C. (eds) Persuasive Technology. Designing for Future Change. PERSUASIVE 2020. Lecture Notes in Computer Science(), vol 12064. Springer, Cham (2020) https://doi.org/10.1007/978-3-030-45712-9 7

- Angulo, C., Pfeiffer, S., Tellez, R., Alenyà, G. Evaluating the use of robots to enlarge AAL services. Journal of Ambient Intelligence and Smart Environments 7(3), pp. 301-313 (2015) https://doi.org/10.3233/AIS-150315
- Hall, A., Wilson, C.B., Stanmore, E., Todd, C. Implementing monitoring technologies in care homes for people with dementia: A qualitative exploration using Normalization Process Theory. International Journal434 of Nursing Studies 2017, 72, 60–70
- Bui, H.D., Chong, N.Y. An Integrated Approach to Human-Robot-Smart Environment Interaction Interface for Ambient Assisted Living. 2018 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO),545 2018, pp. 32–37
- Wengefeld, T., Schuetz, B., Girdziunaite, G., Scheidig, A., Gross, H.M. The MOR-PHIA Project: First Results of a Long-Term User Study in an Elderly Care Scenario from Robotic Point of View. Int. Symposium on Robotics (ISR Europe), Munich, Germany. VDE, 2022, pp. 1–8
- Abdi, J.; Al-Hindawi, A.; Ng, T.; Vizcaychipi, M.P.: Scoping review on the use of socially assistive robot technology in elderly care. BMJ Open 2018, 8 (2018) https://doi.org/10.1136/bmjopen-2017-018815
- Wada, K., Shibata, T., Saito, T., Tanie, K. Effects of robot assisted activity to elderly people who stay at a health service facility for the aged. Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453), 2003, Vol. 3, pp. 2847–2852 vol.3
- Ozdemir, D., Cibulka, J., Stepankova, O., Holmerova, I. Design and implementation framework of social assistive robotics for people with dementia - a scoping review. Health and Technology 2021, 11, 367–378
- 12. Coradeschi, S., Cesta, A., Cortellessa, G., Coraci, L., Galindo, C., Gonzalez, J., Karlsson, L., Forsberg, A., Frennert, S., Furfari, F., Loutfi, A., Orlandini, A., Palumbo, F., Pecora, F., von Rump, S., Štimec, A., Ullberg, J., Ötslund, B. GiraffPlus: A System for Monitoring Activities and Physiological Parameters and

Promoting485 Social Interaction for Elderly. In Human-Computer Systems Interaction: Backgrounds and Applications 3; Hippe, Z.S.; Kulikowski, J.L.; Mroczek, T.; Wtorek, J., Eds.; Springer International Publishing: Cham, 2014

- Broxvall, M., Gritti, M., Saffiotti, A., Seo, B.S., Cho, Y.J. PEIS Ecology: integrating robots into smart environments. Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006, pp. 212–218 https://doi.org/10.1109/ROBOT.2006.1641186.479
- Rasch, R., Sprute, D., Pörtner, A., Battermann, S., König, M. Tidy up My Room: Multi-Agent Cooperation for Service Tasks in Smart Environments. J. Ambient Intell. Smart Environ. 2019, 11, 261–275 https://doi.org/10.3233/AIS-190524
- Linner, T.; Güttler, J.; Bock, T.; Georgoulas, C. Assistive robotic micro-rooms for independent living. Automation in construction 2015, 51, 8–22.
- Do, H.M., Pham, M., Sheng, W., Yang, D., Liu, M. RiSH: A robot-integrated smart home for elderly care. Robotics and Autonomous Systems 2018, 101, 74–92 https://doi.org/10.1016/j.robot.2017.12.008
- Mojarad, R., Chibani, A., Attal, F., Khodabandelou, G., Amirat, Y. A hybrid and context-aware framework for normal and abnormal human behavior recognition. Soft Comput. 2023, 28, 4821–4845
- Soldatos, J., Kyriazakos, S., Ziafati, P., Mihovska, A. Securing IoT Applications with Smart Objects: Framework and a Socially Assistive Robots Case Study. Wireless Personal Communications 2021, 117
- Calderita, L.V., Vega, A., Barroso-Ramírez, S., Bustos, P., Núñez, P. Designing a Cyber-Physical System for Ambient Assisted Living: A Use-Case Analysis for Social Robot Navigation in Caregiving Centers. Sensors 2020, 20
- Meyer, S., Fricke, C. Autonomous assitive robots for older people at home: An exploratory study:" He is always there for me-and I for him too". Zeitschrift fur Gerontologie und Geriatrie 2020, 53, 620–629.
- Antonello, M., Carraro, M., Pierobon, M., Menegatti, E. Fast and robust detection of fallen people from a mobile robot. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 4159–4166. https://doi.org/10.1109/IROS.2017.8206276.606
- Hanheide, M.; Hebesberger, D.; Krajnik, T. The When, Where, and How: An Adaptive Robotic Info-Terminal for Care Home Residents - A Long-Term Study. 2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 2017, pp. 341–34
- Embarak, F., Ismail, N.A., Othman, S. A systematic literature review: the role of assistive technology in supporting elderly social interaction with their online community. Journal of Ambient Intelligence and 611 Humanized Computing 2021, Vol. 13, 1–14. doi:10.1007/s12652-020-02420-1.
- Gulzar, H., Shakeel, M., Itoyama, K., Nakadai, K., Nishida, K., Amano, H., Eda, T. FPGA based Power-Efficient Edge Server to Accelerate Speech Interface for Socially Assistive Robotics. 2023 IEEE/SICE International Symposium on System Integration (SII), 2023, pp. 1–6
- Bustos, P., Manso, L.J., Bandera, A., Bandera, J.P., García-Varea, I., Martínez-Gómez, J. The CORTEX cognitive robotics architecture: Use cases. Cognitive Systems Research 55, 107-123 (2019), https://doi.org/10.1016/j.cogsys.2019.01.003
- Shapiro, M., Preguiça, N., Baquero, C., Zawirski, M. A comprehensive study of Convergent and Commutative Replicated Data Types. https://inria.hal.science/inria-00555588, Research Report RR-7506, Inria, France, 2011.

- 27. Bustos, P., García, J.C., Cintas, R., Martirena, E., Bachiller, P., Núñez, P., Bandera A. DSRd: A Proposal for a Low-Latency, Distributed Working Memory for CORTEX. In: Bergasa, L.M., Ocaña, M., Barea, R., López-Guillén, E., Revenga, P. (eds) Advances in Physical Agents II. WAF 2020. Advances in Intelligent Systems and Computing, vol 1285. Springer, Cham (2021) https://doi.org/10.1007/978-3-030-62579-5_8
- Romero-Garcés, A.; Bandera, J.P.; Marfil, R.; González-García, M.; Bandera, A. CLARA: Building a Socially Assistive Robot to Interact with Elderly People. Designs 2022, 6, 125. https://doi.org/10.3390/designs6060125
- 29. Satoshi Tanaka. DepthAnything-ROS is ROS 2 wrapper for Depth-Anything (2024)
- 30. Yang, Lihe and Kang, Bingyi and Huang, Zilong and Xu, Xiaogang and Feng, Jiashi and Zhao, Hengshuang. Depth Anything: Unleashing the Power of Large-Scale Unlabeled Data (2024)

Integration of a Sign Language Interface in a Socially Assistive Robot

Sohaila Chtioui Bouhou, Alberto Tudela, José Galeas, Óscar Pons, Antonio Bandera, and Adrián Romero-Garcés

Avispa Group, ETSI Telecomunicación, Universidad de Málaga, Málaga-Spain {ajtudela,jgaleas,opfernandez,ajbandera,argarces}@uma.es

Abstract. This work aims to design and implement a Sign Language interface to improve the communication of deaf people in hospital and telemedical environments. Using Mediapipe to detect landmarks and a Random Forest classifier to interpret gestures, a system capable of recognising and translating signs in real time has been developed. The interface includes an intuitive graphical application developed in Tkinter that allows the visualisation of the recognised signs and facilitates interaction with the user. In addition, detection modes for both letters and numbers have been implemented, and mechanisms for remote communication and system updating have been considered. The interface was integrated into a prototype robot to test its functionality in real environments.

Keywords: Sign Language · Socially Assistive Robot · Random Forest

1 Introduction

The World Federation of the Deaf $(WFD)^1$ estimates that around 70 million people worldwide face the reality of deafness, a figure that highlights the importance of improving communication and quality of life for this population. These people are also, in many cases, unable to learn to speak properly. For them, the only form of communication is usually the use of Sign Language. Briefly, a Sign Language is the one that uses the visual-manual modality to convey meaning, instead of employing spoken words. With the aim of teaching children Sign Language, the Istanbul Technical University (ITU) launched the 'Robotic Sign Language, while playing with them [2, 3]. More than 100 children of 3.5-15 years experienced the robots and the games. A Robovie R3 Humanoid Robot and a Nao H25 Humanoid Robot were employed for conducting the experiments. The project is very relevant and interesting, but the focus was on teaching Sign Language. There is a proposal for sign language recognition.

There is currently a significant progress in the research and development of assistive robotics for the elderly or for people with certain disabilities. The presence of these robots is expected to be one of the solutions to attend to this

¹ https://wfdeaf.org/

increasingly numerous segment of the population. In this context, the idea of designing a Sign Language communication interface arises. This can be a crucial tool to facilitate the interaction and communication of deaf people in different environments. Moreover, taking into account the possible penetration of the so-called assistive robots in nursing homes or care centres [4], the design and implementation of this interface was carried out considering its integration in a care robot: the CLARA robot [5]. The aim is to address the additional difficulties faced by deaf people when moving to a care environment. The implementation of this interface would not only enhance communication, but also contribute to improving the medical experience and care for this population group.

Hence, this article describes the design and implementation of a Sign Language interface, whose purpose is to improve the communication and experience of deaf people in care and telemedical environments. Although the development of the interface is the core of this work, we have also addressed the integration within the software architecture of the CLARA robot. The interaction between robot and user is encoded into a Behaviour Tree, and makes use of the software agents included in the architecture for navigation, speech, person detection and identification, or for managing the touchscreen in the torso of the robot [5]. It is important to note that this work is still at a preliminary stage. It will be necessary to test it with a significant group of users to determine its robustness, in a process in which it may be necessary to redesign parts of the framework, especially those related to user interaction.

The rest of the article is organised as follows: Section 2 briefly introduces the state of the art in the topic. Section 3 describes the design and implementation of the proposed solution. Section 4 presents some experimental tests. Finally, Conclusions and future work are drawn in Section 5.

2 Related work

In the field of the design of specific interfaces for sign language communication, there has been a variety of work exploring techniques and technologies. The goal is to improve the accessibility and effectiveness of communication between people with hearing impairments and the rest of society. In this brief evaluation of the state-of-the-art, we have chosen papers that offer different perspectives on how to address the challenges associated with the sign language interpretation and translation. In addition to advances in automatic sign language recognition, other related areas have been explored that contribute to the design of more effective, user-centred interfaces.

In the proposal by Johnny and Nirmala [6], the development of an interface for the automatic recognition of sign language was carried out. This project focused on the implementation of a system that combined signal processing and machine learning techniques to interpret sign language gestures made by users. The main goal of the interface was to convert these gestures into text or speech, thus facilitating communication between hearing impaired people and those who do not know sign language. To avoid the problems associated to using vision, a Fifth Dimension Technologies (5DT) glove is employed. Different machine learning techniques such as neural networks, decision tree classifier, and k-nearest neighbours were evaluated.

The work of von Agris et al. [1] focused on reviewing advances in visual sign language recognition. This study examined different approaches and techniques used in gesture recognition systems, with the aim of better understanding their application in assistive communication environments for people with hearing impairment. Methods based on image processing and machine learning were explored to effectively interpret and translate gestures. However, one of the limitations identified was the difficulty in recognising complex gestures and individual variations in sign language, which could affect the accuracy and reliability of the system in real-world situations.

In addition to these works, other techniques and approaches have been explored in the design of interfaces for sign language communication. For example, some researchers have investigated the use of augmented and virtual reality technologies to enhance user experience and interface interaction [7]. These technologies offer new ways of representing and visualising sign language, which can make it easier for unfamiliar people to understand and learn. These promising approaches open up new possibilities for designing more effective and accessible interfaces for people with hearing impairment.

Previous work in this field has laid the foundation for future research and development in the design of interfaces for sign language communication. Although significant progress has been made, there are still important challenges that need to be addressed in order to create more effective and accessible interfaces that improve communication and inclusion of hearing impaired people in society. In this work we have focused on the design of an interface that allows the recognition of the signs that make up the vocabulary of sign language. This interface should use sensors and interaction devices that we can embark on a robot (camera, speakers, touchscreen). In addition, this work has also addressed the complete integration in the robot, considering that it must be able to interact with the user autonomously.

3 Design and implementation

3.1 Overview of the framework

We can divided the developed framework into two fundamental phases. In the Training Phase, the aim was to create a model capable of recognising hand gestures that represent letters and numbers. This phase involved the collection and processing of hand data, followed by the training of the model using this data. On the other hand, the aim of the Recognition Phase was to identify the gestures made in real time, comparing the data sequences with the previously trained gesture models. Figure 1 illustrates the complete process required to detect the gestures, dividing the system into the two described phases: the Training Phase and the Recognition Phase.



Fig. 1. Overview of the proposed framework

Training Phase In this phase, several images of the hand performing different gestures corresponding to letters and numbers of the alphabet were captured. The goal of this first step was to create a database that allowed the subsequent training of the classifier. To capture the images, a cheap webcam was used. The information stored in the database was processed, for obtaining the relevant features that were used to train the machine learning model. Images of the letters of the Spanish Sign Language (LSE) alphabet were collected.

Specifically, the extracted features were the coordinates of several key points located on the hands. These coordinates were processed to normalise them by subtracting the minimum value from the X and Y coordinates, thus generating a dataset representing the shape and position of the hand in each image. This data was stored together with the labels corresponding to each letter/number. The MediaPipe Hands from Google was used to conduct this step. Further details are provided in Section 3.2.

The Random Forest classifier has been used to implement gesture recognition. This algorithm is briefly described in Section 3.3. Once a model is available, it is saved so that it can be used at run time.

Recognition Phase In this phase, practically the same algorithms used in the Training phase are executed. Thus, the image should be captured, the features should be extracted, and the sign should be recognised using the model already generated. Figure 2 shows a capture of a gesture performed with both hands. The extracted features are shown on the captured image.



Fig. 2. Hands key-points drawn over a captured image

The following sections provide more detail on the more relevant modules of the framework described.

3.2 Hands detection

MediaPipe is an open source platform developed by Google, designed to build computer vision and machine learning solutions efficiently and in real time. It simplifies the creation of complex applications through the use of computational graphs that allow the integration of different data processing modules. The following is a detailed explanation of how MediaPipe works and one of its most outstanding components: Mediapipe Hands².

Mediapipe Hands is one of the most widely used Mediapipe modules. It is specifically designed to detect and track hands and fingers in images and videos. This component is critical for applications that require gestural interaction, such as gesture-based game controllers, virtual and augmented reality, and sign language translation.

Mediapipe Hands follows a two-stage approach to detect and track hands:

- 1. Hand Detection. First, it uses a neural network model to detect the presence of hands in the image or video. This model generates a bounding box around each detected hand.
- 2. Landmark Prediction. Once the hand is detected, another neural network model predicts a set of key-points or 'landmarks' on each hand. These landmarks represent the positions of the joints and ends of the fingers, providing a detailed description of the hand posture.

 $^{^2}$ https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker?hl=es-419



Fig. 3. MediaPipe Hands landmark model

The 21 Mediapipe Hands landmarks are specific points on the hand that indicate the exact position of the joints and ends of the fingers. These points are essential for understanding and analysing the configuration and movements of the hand. Figure 3 shows an image of the landmarks used in Mediapipe.

3.3 Signs recognition

Random Forest is a powerful machine learning algorithm that is widely used in classification and regression problems. This method is based on building multiple decision trees during training and obtaining the final prediction by aggregating the results of these trees. It has been recently used in the framework of sign recognition [8]. Random Forest consists of the following main steps:

- 1. The algorithm creates multiple subsets of training data by sampling with replacement. This means that some data may be selected more than once, while others may not be selected at all.
- 2. For each subset of data, a decision tree is built. During the building of each tree, at each decision node, a random subset of features is considered instead of all available features. This introduces additional variability between trees.
- 3. Once all trees have been built, a prediction is made for each tree. For classification problems, the majority prediction (the class with the most votes) is taken as the final prediction for the forest. For regression problems, the average of the predictions of all trees is taken.

Figure 4 shows how the classifier works and how it manages to arrive at the final prediction.



Fig. 4. Decision tree of the Random Forest classifier

4 Experimental results

4.1 The CORTEX software architecture

The developed framework has been integrated into the CORTEX software architecture [9]. CORTEX is a multi-agent architecture designed to facilitate the creation of information flows among different types of memories and modules. These flows are pumped by agents. Agents are represented as semicircular forms connected to the working memory W, the so-called Deep State Representation (DSR). They can edit it to create and maintain an updated context accessible to the rest of the agents. W is a distributed graph structure implemented using conflict-free replicated data types (CRDT) [10]. Its nodes are elements of a predefined ontology, and the edges can encode geometric transformation or logic predicates. Both nodes and edges can store a list of attributes of a predefined type. In this paper, we use a simplified version of the CORTEX robotics cognitive architecture [9].

4.2 Integration of the framework in the CLARA robot

The Sign Language framework has been successfully integrated in the CLARA robot. The interaction with the person using Sign Language has been encoded using a Behaviour Tree³. Figure 7 shows the complete structure of the Behaviour Tree. In its current version, it only recognises one sign. Briefly, an execution of this use case consists of the following steps:

- The robot approaches to the person to start the use case from a correct interaction distance (this distance can be personalised to each person).

³ https://github.com/BehaviorTree/BehaviorTree.CPP



Fig. 5. CLARA interacting with an user through the Sign Language interface

- The BT encodes two 'say' commands: one to greet the person with its name (this is an attribute of the person node of the DSR), and another one to inform the person that it is going to start the gesture recognition.
- The Gesture Recognition Interface is launched in the touchscreen in the torso of CLARA to start detecting the letters.
- Once the finish button is pressed in the interface, the detected word is saved as a 'word' attribute inside the person node that is interacting with the robot.
- Finally, a 'say' action is executed in which the robot says by voice the detected word through the interface (it reads the value of the word attribute previously updated in the DSR node).
- The use case ends and the robot starts a new task (wandering by default).

Figure 5 shows an user interacting with CLARA using Sign Language. When the sign is recognised, it is added to the person node. Figure 6 shows a snapshot of the DSR after recognising one sign. The person node has been updated.

5 Conclusions and future work

This paper describes the design of a framework for Sign Language recognition implemented using MediaPipe Hands and the Random Forest classifier. The framework has been integrated into the embedded software architecture of a social robot. In order to enable the robot to autonomously execute the process of interacting with a person who communicates using signs, the interaction process has been coded as a Behaviour Tree. This work is only a first approach to the problem. Being integrated in a robot, the final result can be endowed with a great deal of proactivity: if the person's hand is not well detected, the robot could suggest to the person to change its position in the environment; the person could be asked to confirm if the sign has been well recognised when in doubt; and so on.



Fig. 6. A snapshot of the DSR after recognising a sign

As already mentioned in the Introduction, both the framework and the integration in the robot require an evaluation process with real users, since at the moment we have only tested it in the Laboratory and with users who already knew the framework. Extensive testing with a broader demographic and in more varied environments is necessary to truly validate the system's effectiveness and robustness. Obviously, the evaluation with real users will involve changes in the different parts that make up both the recognition framework and the interaction procedure.

Acknowledgments. This work is granted by PDC2022-133597-C42, TED2021-131739B-C21 and PID2022-137344OB-C32, funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGeneration EU/PRTR (for the first two grants), and "ERDF A way of making Europe" (for the third grant).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

 von Agris, U., Zieren, J., Canzler, U. et al. Recent developments in visual sign language recognition. Univ Access Inf Soc 6, 323–362 (2008). https://doi.org/10.1007/s10209-007-0104-x
- Köse, H., Uluer, P., Akalın, N. et al. The Effect of Embodiment in Sign Language Tutoring with Assistive Humanoid Robots. Int J of Soc Robotics 7, 537–548 (2015) https://doi.org/10.1007/s12369-015-0311-1
- Uluer, P., Akalın, N., Köse, H. A New Robotic Platform for Sign Language Tutoring. Int J of Soc Robotics 7, 571–585 (2015) https://doi.org/10.1007/s12369-015-0307-x
- Ohneberg, C., Stöbich, N., Warmbein, A. et al. Assistive robotic systems in nursing care: a scoping review. BMC Nurs 22, 72 (2023). https://doi.org/10.1186/s12912-023-01230-y
- Romero-Garcés, A.; Bandera, J.P.; Marfil, R.; González-García, M.; Bandera, A. CLARA: Building a Socially Assistive Robot to Interact with Elderly People. Designs 2022, 6, 125. https://doi.org/10.3390/designs6060125
- Johnny, S., Nirmala, S.J. Sign Language Translator Using Machine Learning. SN COMPUT. SCI. 3, 36 (2022) https://doi.org/10.1007/s42979-021-00896-y
- Quesada, L., López, G., Guerrero, L. Automatic recognition of the American sign language fingerspelling alphabet to assist people living with speech or hearing impairments. J Ambient Intell Human Comput 8, 625–635 (2017) https://doi.org/10.1007/s12652-017-0475-7
- Ewe E.L.R., Lee C.P., Lim K.M., Kwek L.C., Alqahtani A. LAVRF: Sign language recognition via Lightweight Attentive VGG16 with Random Forest. PLoS ONE 19(4): e0298699 (2024) https://doi.org/10.1371/journal.pone.0298699
- Bustos, P., Manso, L.J., Bandera, A., Bandera, J.P., García-Varea, I., Martínez-Gómez, J. The CORTEX cognitive robotics architecture: Use cases. Cognitive Systems Research 55, 107-123 (2019), https://doi.org/10.1016/j.cogsys.2019.01.003
- Shapiro, M., Preguiça, N., Baquero, C., Zawirski, M. A comprehensive study of Convergent and Commutative Replicated Data Types. https://inria.hal.science/inria-00555588, Research Report RR-7506, Inria, France, 2011



Fig. 7. Behaviour Tree responsible of managing the interaction using the developed Sign Language Interface

Impact of Real time on Active Perception Systems applied to Social Robotics

A. Alberto García¹[0000-0002-3498-2618]</sup>, Juan Diego Peña Narváez¹[0009-0008-2472-7091]</sup>, Rodrigo Pérez Rodríguez²[0000-0003-1629-0973]</sup>, José Miguel Guerrero Hernández²[0000-0003-2521-514X]</sup>, Francisco Martín Rico²[0000-0003-312-1-5744]</sup>, and Juan Carlos Manzanares Serrano²[0009-0008-7122-1165]

¹ Escuela de Doctorado Rey Juan Carlos University, calle Tulipán, Móstoles, Spain ² Intelligent Robotics Lab, Universidad Rey Juan Carlos, Fuenlabrada, 28943, Spain {aa.garciag, jd.pena.2023 }@alumnos.urjc.es {rodrigo.perez, josemiguel.guerrero, francisco.rico, juancarlos.serrano }@urjc.es https://intelligentroboticslab.gsyc.urjc.es/

Abstract. Real-time capabilities are crucial for enabling robots to interact effectively in dynamic human environments, addressing latency and computational constraints that hinder traditional systems. This paper examines the essential role of real-time processing in active perception systems within social robotics. We propose an integrated approach within the ROS 2 framework, leveraging advanced object detection models and cascade lifecycle nodes to ensure robust and efficient tracking of individuals and objects. The robot's head or camera is required to move and direct itself toward visual stimuli. Our experimental validation demonstrates significant improvements in orientation error rates with real-time configurations, particularly under high-stress scenarios. The findings highlight the practical advantages of real-time systems in enhancing situational awareness and interaction quality in social robotics.

Keywords: Open Source Robotics · Perception Systems · Social Robotics · Open Frameworks · Real-Time Processing.

1 INTRODUCTION

In the rapidly evolving field of social robotics, active perception systems have become a cornerstone for enabling robots to interact effectively and intuitively within human environments [3]. Real-time processing is critical for these systems, as it allows robots to adapt to dynamic changes and respond promptly to human behaviors and environmental cues [1]. This paper, investigates the essential role of real-time capabilities in enhancing the performance and efficacy of active perception systems within social robots.

The motivation for this paper arises from the need to address the challenges posed by increased system load on the effectiveness of tracking individuals or objects. In dynamic environments, where timely and accurate perception is paramount, traditional systems often struggle with latency and computational constraints. We propose that incorporating real-time processing can ensure reliable and efficient tracking, even as system demands escalate, thereby improving robots' situational awareness and interaction quality.

Robocup is a major international initiative promoting advancements in autonomous robotics through competitive events. Featuring leagues such as Robo-CupSoccer, RobocupRescue, RoboCup@Home, RoboCupJunior, and RoboCup Industrial. [23]

The RoboCup@Home league aspires to advance service and assistive robotic technology for future domestic applications. The evaluation of the robots' skills and performance is conducted through a series of benchmark tests in realistic, non-standardized home environments. RoboCup@Home serves as a valid testbed due to its rigorous evaluation of service and assistive robotic technologies in realistic, non-standardized home environments. [22]

The significance of focusing on RoboCup tests lies in their comprehensive and practical approach to assessing robotic capabilities across various domains, including Human-Robot Interaction, Navigation, and Object Recognition. The necessity for real-time attention during these tests is paramount, as it ensures the robots can effectively adapt to dynamic conditions and interact seamlessly with humans, thereby demonstrating their potential for future domestic applications.

The necessity for real-time attention systems is crucial for effective humanrobot interaction and for the robots' ability to follow and assist individuals. Realtime attention enables robots to respond promptly and accurately to dynamic changes in the environment and human behavior. This capability is essential for ensuring seamless interaction, as it allows robots to anticipate and react to human actions, maintain safety, and provide timely assistance. Additionally, realtime attention systems enhance the robot's ability to perform complex tasks such as tracking a person through a cluttered environment, recognizing and adapting to human gestures, and making split-second decisions that improve the overall user experience and reliability of the robotic system.

ROS, as described by Quigley et al. [24], has established itself as the leading framework in robotics, offering a structured communication layer above heterogeneous compute clusters rather than a traditional operating system. ROS addresses the complexities of robotics software integration, enabling code reuse across diverse hardware platforms.

ROS 2 has emerged as a significant advancement over its predecessor, addressing many of the critical limitations identified in ROS 1. While ROS 1 laid the foundation for robotics research with its modular framework and extensive utilities, it struggled with issues related to security, network topology, and scalability, particularly in non-traditional and large-scale embedded systems. ROS 2 also brings enhanced real-time capabilities, ensuring more reliable and predictable performance for time-sensitive applications. This redesign facilitates better reliability and performance, making ROS 2 well-suited for both research and commercial deployment across diverse environments. [25] Usually, real-time considerations are not a primary focus in high-level applications. However, addressing real-time requirements is becoming increasingly relevant in modern robotic systems. In our forthcoming paper, we will elaborate on strategies specifically tailored to meet these demands, thereby contributing valuable insights to optimizing real-time performance in advanced robotic applications

This paper is structured as follows: Section II reviews related work, providing a context for our contributions and highlighting the advancements and limitations in current active perception systems. Section III outlines the general architecture of our proposed system, detailing the key components and their interactions. In Section IV, we present the strategies employed within the ROS 2 framework to achieve real-time performance, focusing on the communication mechanisms between the perception and attention systems. Section V offers an experimental validation of our approach, showcasing its effectiveness through various scenarios and under varying system loads. Finally, Section VI concludes the paper, summarizing our findings and discussing potential avenues for future research, while Section VII addresses the limitations of our current implementation and suggests directions for future improvements.

2 RELATED WORK

Real-time perception systems in robotics have been a focal point of research, with various methodologies developed to enhance a robot's ability to interact dynamically with its environment. The study "Real-Time Multiple Human Perception With Color-Depth Cameras on a Mobile Robot" [1] explores a system that employs color-depth cameras to detect and track multiple humans in realtime. This approach underscores the importance of integrating robust algorithms capable of processing sensory data swiftly, which is essential for maintaining accurate tracking in cluttered environments and ensuring effective human-robot interaction.

Active perception systems represent a significant advancement in robotic perception, allowing robots to actively control their sensors to gain a better understanding of their surroundings. In "Visual Attention in a Mobile Robot" [2], the authors present a method by which a mobile robot dynamically adjusts its focus based on areas of interest within its field of view. This system improves the efficiency and quality of robot-environment interactions by optimizing the use of processing resources and concentrating on relevant stimuli, which is critical for applications like surveillance and autonomous navigation.

Attention mechanisms in robotics, inspired by human cognitive processes, are becoming integral to enhancing a robot's perceptual capabilities. Authors in [4] discuss the implementation of attention models that allow robots to prioritize sensory inputs, improving their efficiency and responsiveness.

These mechanisms are particularly beneficial in complex environments where robots must discern and react to multiple stimuli simultaneously, significantly enhancing a robot's ability to perform tasks such as object recognition, navigation, and human interaction. In [2] authors utilized segmentation directly to focus a robot's attention. [4] advanced this by proposing a robot scanning method based on human visual search models, optimizing long-term information acquisition about target locations. [5] integrated peripheral-foveal vision with an attention mechanism to enable the "Curious George" robot to build a semantic map of its environment, effectively labeling objects.

In other work, Kismet was developed, a robotic head using various filters like face detection, color, and movement to direct its gaze towards salient areas [6]. Also, [6] introduced the WABIAN humanoid robot with an active vision system guided by visual and auditory inputs, inspired by Wolfe's research on identifying relevant image areas. Numerous studies have focused on salience, emphasizing image region relevance without spatial information beyond the image itself [8],[9],[10],[11],[12].

Other notable contributions include introducing "Regions of Interest" in images to guide a robot's camera, based on the robot's active tasks [15]. Authors in[16] combined bio-inspired models with neural networks to create saliency maps focusing on spatial areas within the environment. Integrating attention systems into cognitive architectures,[17] proposed a method for managing visual attention in humanoid robots within a cognitive architecture based on salience. [18] discussed Cortex, a cognitive architecture that directs the robot's gaze to areas likely containing items.[14] introduced the EgoSphere concept in the iCub robot, integrating sensory fusion to adjust salience. [19] demonstrated a social robot using visual attention to control gaze direction within group conversations, integrating stimuli such as people's locations and sound sources.[20]described an attention system that dynamically manages a robot's focus on various stimuli through a client-server architecture, ensuring efficient visual attention management.

3 GENERAL ARCHITECTURE DESCRIPTION

The proposed architecture, depicted in Figure 1 comprises multiple interconnected systems, including the perception system, the attention system, and the dialogue system, each enhancing the robot's functionality and effectiveness in social environments.



Fig. 1: Diagram of the Proposed Social Robot Architecture

- Tiago Robot:

- Nav2 and MoveIt2 are integral components of the ROS2 software.
- Publishes to:
 - * /head_front_camera/rgb/image_raw
 - * /head_controller/joint_trajectory
- Subscribes to:
 - * /head_controller/joint_trajectory
- Computer:
 - Perception System:
 - * Receives images from Yolo V8 via /input_image.
 - * Publishes detections to perception_system_interfaces/msg/DetectionArray.
 - * Interacts with the attention system to identify individuals through the TF System.
 - Attention System:
 - $* \ Issues attention \ commands \ via \ \verb+attention_system_msgs/msg/AttentionCommand.$
 - * Sends joint trajectory commands through control_msgs/action/FollowJointTrajectory.

- * Seamlessly integrates with the Perception System, dynamically adjusting the robot's focus based on real-time data, ensuring timely and contextually relevant interactions.
- **Dialogue System** facilitates Human-Robot Interaction (HRI), enabling the robot to listen and speak.
- Yolo V8 is used for object and person detection in images.
- The **Transform Tree (TF)** is used for coordinate transformation and spatial referencing between systems.
- The **Behaviour Tree Controller** incorporates behavior tree nodes that enable dynamic and adaptable robotic behaviors within social environments.

The perception system acquires and processes environmental information through various sensors, primarily focusing on camera data. This information is crucial for the robot's interaction and decision-making processes.

Playing a pivotal role in social robotics, the attention system dynamically focuses on specific entities or individuals within its environment. It manages and prioritizes the robot's focus on a single stimulus, ensuring efficient and contextually appropriate interactions. A crucial aspect of the attention system is its integration with the perception system. The perception system generates a transform (tf) for the detected person, encapsulating their position and orientation in the environment. By subscribing to this single transform, the attention system can dynamically track and follow the person, maintaining continuous focus and enhancing interaction capabilities.

Seamless integration between the perception and attention systems allows the robot to adjust its focus dynamically based on real-time data, ensuring interactions are timely and contextually relevant.

Behavior tree nodes incorporated in the Behaviour Tree Controller enable dynamic and adaptable robotic behaviors within social environments. These nodes provide a flexible and modular framework for defining and executing complex robotic tasks. An example behavior tree diagram (Figure 2) illustrates the implementation and functionality of these nodes within our system.



Fig. 2: Diagram of a Simple Behaviour Tree from A Concise Introduction to Robot Programming with ROS2 [26]

A state machine is implemented through Lifecycle nodes, ensuring deterministic program management and reliable execution. Node states can depend on other nodes, creating a cascading startup where each node activates only after its dependencies are active [21]. This approach offers significant advantages, including predictability in parameter access during the configuration transition, coordination of startup sequences among multiple nodes, and programmatic control beyond traditional constructors for component initialization.



Fig. 3: Diagram of Lifecycle Nodes: States and Transitions from A Concise Introduction to Robot Programming with ROS2 [26]

Lifecycle nodes in a robotic system can be delineated into several primary states, as depicted in Figure 3:

- Unconfigured: The node is not yet initialized and cannot perform any meaningful operations.
- Inactive: Once configured, the node is initialized but not yet performing its primary functions, effectively on standby.
- Active: The node is fully operational and performing its designated functions.
- Finalized: The node has been decommissioned and is safely stopped.

In ROS2, nodes progress through various states, each facilitating a different phase of the node's lifecycle. Initially, a node must trigger the configure transition to enter the Inactive state. From the Inactive state, it can transition through the activate transition to become Active. Conversely, it is possible to transition from the Active to Inactive state through the deactivate transition. In the event of an error, the node can move to the Finalized state. When a node has completed its task, it can transition to the Finalized state. Each state transition allows for the execution of necessary tasks and checks. A transition may fail if the conditions specified in the transition code are not met.

4 ROS 2 Strategies for Real-Time Processing and Active Perception

Real-time systems as shown in Figure 4, are designed to respond to stimuli within strict time constraints, ensuring timely and predictable interactions. These systems are characterized by their ability to meet predefined deadlines for task execution, which is crucial for applications requiring consistent and immediate responses. In the context of robotics, real-time systems are integral for maintaining operational reliability and safety, especially in dynamic environments where latency can lead to significant consequences.



Fig. 4: Instances of Real-Time Systems from Real-time programming with ROS 2 [27]

In the robotics field, these time-critical systems must adhere to several essential requirements to ensure they meet the necessary performance and reliability standards. Bounded response time, as can be observed in Figure 5a, refers to the system's capability to respond to an input or event within a fixed and predictable timeframe. This is crucial in ensuring that the robot can react promptly to changes in its environment, such as avoiding obstacles or interacting with humans. Delayed responses could lead to operational failures or safety hazards. This concept is typically illustrated by a callback function that must complete its execution within a specific time limit, demonstrating the bounded response time. High-frequency control loops, operating at approximately 1000Hz, are essential to maintain stability and prevent unsafe conditions. These control loops are fundamental in managing the dynamic behavior of robots, where an unstable controller could result in hazardous situations. Moreover, object detection and collision avoidance mechanisms, functioning at around 100Hz, are critical for ensuring that the robot can detect and avoid obstacles in a timely manner, preventing collisions.

Guaranteed end-to-end latency, depicted in Figure 5b, ensures that the total time taken from the initial input to the final output (end-to-end latency) is consistent and predictable. In a robotic system, this guarantees that complex processes involving multiple steps (e.g., sensing, processing, and acting) are completed within an acceptable time frame. This predictability is essential for synchronized and coordinated actions. An illustration of this concept usually involves a series of interconnected processes or nodes, with the total time from start to end being measured to ensure it remains within the guaranteed limits.

Deterministic update rates, illustrated in Figure 5c, refer to the system's ability to perform regular updates at fixed intervals, without deviation. This consistency is important for tasks that require regular monitoring and updates, such as maintaining balance, tracking moving objects, or continuously processing sensor data. It ensures the robot operates smoothly and reliably. This requirement is often portrayed by multiple processes being executed at regular intervals, highlighting the importance of maintaining a consistent update rate.



(a) Bounded response time Diagram

(b) Guaranteed end-toend latency Diagram

(c) Deterministic end-toend latency Diagram

Fig. 5: Comparison of different latency diagrams from Real-time programming with ROS 2 [27]

We have integrated real-time processing into the perception and attention systems, as presented in the snippet of code 1.1. This integration significantly enhances the robot's ability to interact and respond effectively in social environments. The proposed architecture leverages these real-time capabilities to ensure that the robot can maintain situational awareness and engage in contextually appropriate interactions, thus improving the overall quality of its performance in dynamic settings.

```
rlcpp::ExecutorDptions exe_rt_options;
rlcpp::executors::MultIThreadedExecutor executor_rt(exe_rt_options, 4);
rlcpp::executors::MultIThreadedExecutor executor_rt(exe_rt_options, 4);
rd::make_shared_ptrcplop_cascade_lifecycle:CascadeLifecycleNode>> sched_nodes = {
rd::make_shared_ptrcplin.system::DopleDtectionNode>(node_options),
rd::make_shared<perception.system::DopleCascadeLifecycle:CascadeLifecycleNode>> sched_nodes = {
// Adding systems nodes to the appropriate executor
for (auto & sched_node : ached_nodes) {
executor_rt.add_node(sched_node) {
sched_node<>trigger_tomasite();
}
// Change systems nodes state to Configure
for (auto & sched_node : ached_nodes) {
sched_node<>trigger_transition(lifecycle_msgs::msg::Transition::TRANSITION_CONFIGURE);
}
// Spinning systems nodes in their executors
auto realize_trans achd: ached_sel()
sched_prime sch:
sch.sched_priming (ptread_self(), SCHED_FIFO, &sch) == -1) {
percor("pthread_setachedprima failed");
executor_rt.spin();
}
// clcpp::spin(aux_node);
realize_trans_ion();
}
}
// Spinning realize_trans_ion();
}
// Spinning realize_trans_ion();
}
// Spinning realize_trans_ion();
// Spinning re
```

Listing 1.1: Attention and Perception Systems Configured for Real-Time Operation

In the ROS 2 framework, the interaction between the perception and attention systems is orchestrated through a series of strategies for active perception. These strategies leverage ROS's capabilities to facilitate seamless communication and coordination between the two systems.

Specifically, the perception system in ROS 2 provides transforms representing the detected entities to the attention system. These transforms contain essential information, such as the position and orientation of the entities in the environment. The attention system, in turn, subscribes to these transforms, enabling it to focus on and track the detected entities dynamically.

By utilizing ROS's messaging infrastructure, the perception system can efficiently relay relevant real-time data to the attention system. This integration ensures that the attention system remains synchronized with the perceptual input, enabling the robot to maintain an appropriate and continuous focus on the entities of interest.

The system is based on integrating the YOLO v8 model (You Only Look Once version 8), one of the most advanced architectures for real-time object detection and recognition. A perception listener has been implemented as a single token, requiring specification of the user's interest in a particular object or person. This allows the perception system to efficiently subscribe to and respond to relevant detections, facilitating a swift and appropriate response from the robot to its dynamic environment. This modular and detailed approach ensures that the robot can operate autonomously and effectively in social robotics applications, where interaction with humans and objects is fundamental.

5 EXPERIMENTAL VALIDATION

Our research relies heavily on a geometric transformation framework known as TF, which effectively manages the relationship, encompassing translation and rotation, between frames or reference axes. Through this system, coordinates within one frame can seamlessly transition into another provided that both frames are linked within the same TF tree. This tree structure of TF connections ensures that each frame maintains a singular parent frame while potentially possessing multiple child frames.

The Tiago robot incorporates two pivotal joints tasked with orienting the orbbec astra camera: head_1_link, designated as H1L, facilitating horizontal adjustment (pan), and head_2_link for vertical adjustment (tilt). Initially, the transformation matrix was derived from H1L to the individual ($H1L \rightarrow person$), acquired via the perception system, which furnishes the individual's transformation. Subsequently, the error metric that can be observed in Equation 1 (σ) was determined as the absolute value of the arctangent of the ratio of the Y-translation (Y_{trans}) to the X-translation (X_{trans}).

$$\sigma = \tan^{-1} \left(\left| \frac{Y_{\text{trans}}}{X_{\text{trans}}} \right| \right) \tag{1}$$

In order to assess the efficacy of our methodology, a series of experiments were conducted within the Robotics Laboratory as shown in Figure 6).



Fig. 6: Visualization in Rviz depicting YOLOv8 image on the left and transformations of head_1_link and person on the right.

Each experiment lasted approximately 1.5 minutes and investigated six distinct scenarios. These scenarios were classified into Real-Time and Non-Real-Time contexts, each exposed to different levels of CPU stress: low, medium, and high.



Fig. 7: Robotics laboratory test scenario, illustrating the journey from point A to point B: on the left, the individual starting at point A; on the right, the individual arriving at point B.

The experiments employed the Tiago robot³ in conjunction with a computer system featuring an AMD Ryzen 9 7845hx processor with 24 cores, 32 GB of RAM, and an NVIDIA 4060 GPU throughout all trials. To emulate system load, we employed the s-tui package⁴.

The outcomes of these experiments are presented in the subsequent figures 8, 9, and 10. The figures reveal strikingly similar results between low stress (LS) and medium stress (MS) conditions across both real-time and non-real-time setups. However, in high stress (HS) scenarios, notable distinctions emerge. Specifically, during the initial moments of high stress, the error rate nearly doubles in the non-real-time setup compared to the real-time configuration. This underscores the superior efficacy of real-time systems in mitigating errors under high stress conditions.

³ https://pal-robotics.com/es/robots/tiago/

⁴ https://github.com/amanusk/s-tui





Fig. 8: Error Over Time in RT and NRT with low stress CPU load

Fig. 9: Error Over Time in RT and NRT with medium stress CPU load



Fig. 10: Error Over Time in RT and NRT with high stress CPU load

The negligible differences in mean and standard deviation values between the real-time and non-real-time configurations under low stress conditions indicate that the performance benefits of real-time systems are not significantly evident in these scenarios. Both configurations exhibit low error rates and comparable performance metrics.

In the context of medium stress levels, although the differences are minor, the real-time configuration demonstrates a slight improvement in mean error rate, indicating marginally enhanced performance. This suggests that the real-time system may manage medium stress conditions with slightly greater efficiency than the non-real-time system.

In the case of high stress levels, a substantial increase in mean performance values and standard deviations is observed in non-real-time scenarios. This indicates that the system's performance becomes increasingly unpredictable and variable under HS in non-real-time conditions.

The following table1 illustrates a detailed comparison of performance metrics across stress levels and system configurations.

Scenario	Stress Level	Mean	Standard Deviation
Non-Real-Time	Low	0.134	0.075
Real-Time	Low	0.138	0.076
Non-Real-Time	Medium	0.148	0.086
Real-Time	Medium	0.137	0.089
Non-Real-Time	High	0.206	0.195
Real-Time	High	0.176	0.117

Table 1: Summary of Mean and Standard Deviation for Each Experimental Scenario and Stress Level

6 CONCLUSIONS

In this paper, we have explored the critical role of real-time processing in enhancing active perception systems for social robotics. Our investigation focused on integrating real-time capabilities into the perception and attention systems of robots, thereby improving their ability to interact effectively and intuitively in dynamic human environments.

We began by highlighting the challenges posed by latency and computational constraints in traditional perception systems, particularly in scenarios requiring timely and accurate tracking of individuals or objects. By leveraging real-time processing, we demonstrated how these challenges can be mitigated, ensuring robust performance even under increasing system loads.

Through our experimental validation, conducted using the ROS 2 framework and the Tiago robot, we evaluated the effectiveness of our approach across various stress levels and scenarios. The results consistently showed that real-time configurations significantly reduced orientation error rates, particularly under high-stress conditions, highlighting the practical advantages of real-time systems in dynamic environments.

Moreover, our architecture's modular design and integration with ROS 2 facilitated seamless communication between perception and attention systems, enabling the robot to dynamically adjust its focus based on real-time data. This capability is crucial for maintaining contextually appropriate interactions with humans and objects in social settings.

7 LIMITATIONS AND FUTURE WORK

An important consideration in advancing real-time perception systems for social robotics is the dependency on hardware configurations. The efficacy of enhancements in real-time processing capabilities may vary based on factors such as processing power, memory capacity, and sensor integration.

We are actively integrating real-time processing capabilities into our systems specifically for the RoboCup@Home competition. This effort seeks to elevate the performance of the Tiago Robot in competitive settings, where responsiveness and robust real-time decision-making are crucial. By harnessing real-time processing technologies, we anticipate achieving significant advancements in autonomy and adaptability during complex tasks and interactions. Key tasks under consideration include real-time tracking of individuals, precise identification of specific persons within crowded environments, and the interpretation of human gestures. These capabilities are pivotal for enabling our robots to navigate and interact seamlessly in dynamic and unpredictable settings typical of domestic environments encountered in the RoboCup@Home competition.

Additionally, we are extending our real-time processing framework to seamlessly integrate with diverse robotic platforms beyond our current implementation with the Tiago robot. This extension will broaden the applicability of our advancements across various robotic systems, enhancing their ability to operate with heightened responsiveness and reliability in real-world social settings.

In the field of robotics, our findings can contribute to the advancement of robotic perception and interaction capabilities. Enhanced detection and interpretation mechanisms can improve a robot's ability to navigate and interact with dynamic environments. For example, service robots could benefit from improved object recognition and navigation skills, leading to better performance in tasks such as cleaning, delivery, and elder care.

Furthermore, the implications extend to human-computer interaction (HCI). By improving the ways in which systems understand and respond to human inputs, the findings can lead to more intuitive and seamless interfaces. This could revolutionize the way users interact with technology, making it more accessible and user-friendly.

In conclusion, while this study provides valuable insights within a controlled experimental setup, its broader implications and potential applications are vast. By extending these findings to various fields and considering their long-term benefits, we can appreciate the full impact of this research.

Acknowledgments. This study was funded by CoreSense (grant number 10107025).

References

 Zhang, H., Reardon, C., Parker, L.E.: Real-time multiple human perception with color-depth cameras on a mobile robot. IEEE Trans Cybern 43(5), 1429-1441 (2013). https://doi.org/10.1109/TCYB.2013.2275291

- Scheier, C., Egner, S.: Visual attention in a mobile robot. In: ISIE'97 Proceeding of the IEEE International Symposium on Industrial Electronics, vol. 1, pp. SS48-SS52. IEEE (1997). https://doi.org/10.1109/ISIE.1997.651734
- Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 (2020).
- Butko, N., and Movellan, J. (2009). "Optimal scanning for faster object detection," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (Miami, FL), 2751–2758.
- Meger, D., Forssén, P.-E., Lai, K., Helmer, S., McCann, S., Southey, T., et al. (2008). Curious george: an attentive semantic robot. *Rob. Auton. Syst.* 56, 503–511. doi: 10.1016/j.robot.2008.03.008
- Breazeal, C., and Scassellati, B. (1999). "A context-dependent attention system for a social robot," in *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI '99*, (San Francisco, CA: Morgan Kaufmann Publishers Inc.), 1146–1153.
- Hashimoto, S., Narita, S., Kasahara, H., Shirai, K., Kobayashi, T., Takanishi, A., et al. (2002). Humanoid robots in Waseda University—Hadaly-2 and WABIAN. *Auton. Rob.* 12, 25–38. doi: 10.1023/A:1013202723953
- Itti, L., Koch, C., and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* 20, 1254–1259. doi: 10.1109/34.730558
- Hou, X., and Zhang, L. (2007). "Saliency detection: a spectral residual approach," in 2007 IEEE Conference on Computer Vision and Pattern Recognition (Minneapolis, MN), 1–8.
- Harel, J., Koch, C., and Perona, P. (2006). "Graph-based visual saliency," in Proceedings of the 19th International Conference on Neural Information Processing Systems, NIPS'06 (Cambridge, MA: MIT Press), 545–552.
- Goferman, S., Zelnik-Manor, L., and Tal, A. (2012). Context-aware saliency detection. *IEEE Trans. Pattern Anal. Mach. Intell.* 34, 1915–1926. doi: 10.1109/T-PAMI.2011.272
- Grotz, M., Habra, T., Ronsse, R., and Asfour, T. (2017). "Autonomous view selection and gaze stabilization for humanoid robots," in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (Vancouver, BC), 1427–1434.
- García, J. F., Rodríguez, F. J., Martín, F., and Matellán, V. (2010). "Using visual attention in a NAO humanoid to face the RoboCup any-ball challenge," in 5th Workshop on Humanoids Soccer Robots, (Nashville, TN), 1–6.
- 14. J. Ruesch, M. Lopes, A. Bernardino, J. Hornstein, J. Santos-Victor, and R. Pfeifer, Multimodal saliency-based bottom-up attention: a framework for the humanoid robot iCub, in 2008 IEEE International Conference on Robotics and Automation (Pasadena, CA), 2008, pp. 962–967.
- Bachiller, P., Bustos, P., and Manso, L. J. (2008). "Attentional selection for action in mobile robots," in *Advances in Robotics, Automation and Control*, Chapter 7, eds J. Aramburo and A. R. Trevino (Rijeka: IntechOpen), 111–136.
- Stefanov, K., Salvi, G., Kontogiorgos, D., Kjellström, H., and Beskow, J. (2019). Modeling of human visual attention in multiparty open-world dialogues. ACM Trans. Hum. Robot Interact. 8, 8:1–8:21. doi: 10.1145/3323231
- Agüero, C. E., Martín, F., Rubio, L., and Cañas, J. M. (2012). Comparison of smart visual attention mechanisms for humanoid robots. *Int. J. Adv. Rob. Syst.* 9:233. doi: 10.5772/53571

- Manso, L. J., Gutierrez, M. A., Bustos, P., and Bachiller, P. (2018). Integrating planning perception and action for informed object search. *Cogn. Proc.* 19, 285–296. doi: 10.1007/s10339-017-0828-3
- P. Aliasghari, A. Taheri, A. Meghdari, and E. Maghsoodi, *Implementing a gaze control system on a social robot in multi-person interactions*, SN Appl. Sci. 2, 1–13 (2020), doi: 10.1007/s42452-020-2911-0.
- Martín, F., González, A., López, J., Rodríguez, S.: Dynamic Visual Attention System for a Mobile Robot. J Intell Robot Syst 102, 36 (2021).
- Macenski, S., Soragna, A., Carroll, M., Ge, Z.: Impact of ROS 2 Node Composition in Robotic Systems. IEEE Robotics and Automation Letters 8(7), 3996-4003 (2023). https://doi.org/10.1109/LRA.2023.3279614
- RoboCup Federation, RoboCup@Home: Overview and Information, https:// athome.robocup.org/, last accessed 2024/06/14.
- RoboCup Federation, RoboCup 2024: Official Announcement and Overview, https://www.robocup.org/, last accessed 2024/05/27.
- 24. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: an open-source Robot Operating System. ICRA Workshop on Open Source Software, vol. 3, no. 3.2, p. 5 (2009). Kobe, Japan.
- Macenski, S., Foote, T., Gerkey, B., Lalancette, C., Woodall, W.: Robot Operating System 2: Design, architecture, and uses in the wild. Sci Robot 7, 66 (2022). https: //doi.org/10.1126/scirobotics.abm6074
- Rico, F.M.: A Concise Introduction to Robot Programming with ROS2. 1st edn. Chapman and Hall/CRC, New York (2022). https://doi.org/10.1201/ 9781003289623
- 27. S. Wu, J. Staschulat, S. Eng, and O. Bell. (2023). Real-time programming with ROS 2, ROScon 2023 [Workshop]. Available: https://docs.google.com/presentation/ d/1yHaHiukJe-87RhiN8WIkncY23HxFkJynCQ8j3dIFx_w/edit#slide=id.p

Discrete Active Inference with Dynamic Scene Graphs

Pablo Lanillos¹ and Pablo Bustos²

¹ Consejo Superior de Investigaciones Científicas, Instituto Cajal, España p.lanillos@csic.es
² RoboLab, Universidad de Extremadura pbustos@unex.es

Abstract. Active Inference is a cutting-edge theoretical framework that leverages Bayesian inference, control theory, and the free energy principle to explain perception, action, and learning in both biological and artificial systems. This paradigm posits that systems actively infer the causes of their sensory inputs and seek to minimize the surprise between the internal model (learnt by experience) and the actual world state given the observations. However, it usually assumes that the generative model of the world is known—particularly in the discrete domain. Hence, providing grounded and dynamic internal representations of the world can provide the scalability for real world applications, such as robotics. Here, we discuss the potential of integrating dynamic scene graphs, composed of heterogeneous nodes and edges that code parts, objects and individuals in the world with human-shared semantics, with model-based Bayesian inference approaches. We suggest that this type of representation, widely used in Robotics, can be endowed with a kind of dynamics that provides more expressiveness than other solutions such as behaviour trees over fixed ontologies or categories. We present the advantages and the challenges of such a model for robotics through examples framed in real-world use cases.

Keywords: Active Inference, Dynamic Scene Graphs, Robotics

1 Introduction

Most contemporary Robotics Cognitive Architectures (RCA) rely on a working memory (WM) to represent the current state of the robot and its surrounding environment. This representation is usually built from inputs coming from different perceptual modalities that are processed to detect and stabilize objects in the vicinity relevant to the task at hand. A common software implementation is a scene graph, where vertices represent detected objects and edges encode relationships among them. Scene graphs have their origin in computer graphics, where acyclic graphs (DAG) are used to represent a scene by the topological and spatial relations among objects [28]. In computer vision, scene graphs have been widely used to transform visual scenes into semantically structured representations, usually expanding into multi-layer structures covering large space areas. Due to the complexity of creating these structures, many efforts have been directed towards automating this process using deep neural networks trained with large datasets [7][34]. In this context, dynamic scene graphs have been defined as an extension to static graphs that capture the dynamic relationships among objects that appear in video sequences [9][30]. In the rest of this paper we will use the term dynamic scene graph (DSG) in the sense of graphs that change in time to adapt to the static or dynamic content of the world.

In robotics, scene graphs were introduced in [2], with some precedents based on related concepts [28][23][11] [33]. The first results in the semi-automated construction of large scene graphs were reported in [1][16] and later, a series of papers by another group presented successive results that culminated in the Hydra architecture [1] [27] [13] [8] [14] These systems have been designed as tools to create structured representations of large scenes and include all detectable elements and relationships in the environment. They start from a metric reconstruction and build on it to extract the human-shared concepts (e.g., rooms, furniture, and people) represented in the graph. The goal is to create the scene graph per se, not to make it a part of what a robot would need to succeed in its tasks.

Although the results presented in these architectures are impressive in terms of recovered detail and low error rates on the datasets, we are interested in a type of model that could support real-time discrete active inference in robots. These graphs and especially the processes supporting them within a robotics control architecture must meet some specific constraints that we will explore in the next sections.

One of the most influential approaches in modern robotics is to model perception, planning, and control as a Markov decision problem and use Bayesian modeling to solve it. We can see this in several areas of research from classical estimation to deep reinforcement learning. The difference between the methods is usually on how we model the problem and how we cast it to make it tractable (e.g., variational methods, optimization, etc).

Here we address a specific type of approximate Bayesian inference, named *Active Inference (AIF)*, which originated in computational neuroscience [24], but that has been recently investigated in robotics and artificial intelligence [19]. In this mathematical framework, the agent/robot has a probabilistic model of the world (generative or world model) [29] and perceives and plans to minimize surprise or the discrepancy between the predicted observations and the data being observed. Particularly in high-level decision making or planning, this internal model has to be designed (through factorization of the variables) or learned through experience. In the case of discrete state representation, we face the unsolved challenge of grounding representations. Therefore, in the majority of the works in the literature the structure of the states, likelihoods, and transition functions are provided. For instance, by defining the categorical distributions and the matrix sizes. Recently, some effort has been made to connect estimation and

planning to behavior or semantic trees [26]. Thus, enriching the representation power for robotic tasks. In a real-world situation, we need dynamic representation structures that allow us to flexibly represent the semantics of the world but at the same time, to provide inference about the current state of the robot and the best plan to execute.

2 Generative models in AIF

In the same way that we can use a graphical representation to evaluate the possible plans, dynamic scene graphs can represent the current model of the world that can be used to, for instance, infer the robot or any other element's position in the environment. This scene graph has to account for the continuous changes in the environment and in the robot itself, acting as an effective working memory. DSG, thus, has the potential to be used as an internal discrete world model. However, to make DSG a proper generative model, we need to find a sound mapping between the DSG's creating and updating mechanisms and the generative model dynamics.

In the Active Inference framework, an agent that does not have access to the world/body state has to infer it from the sensor measurements. In terms of Bayesian inference, it infers the most probable state of the world x using imperfect noisy sensory observations y and prior information—for instance, learnt by experience. According to Bayes rule, the probability of a state x given the observed data y is encoded in the posterior probability p(x|y).

The agent aims to approximate its internal model to the true (posterior) distribution given the sensory inputs, the current hidden states, and action policies. By reducing the difference between the predicted sensory inputs and the actual sensory inputs from the environment, the agent can revise its beliefs and enhance its internal model to better reflect the underlying structure of the world. This process allows the agent to make more accurate predictions and choose actions that are effective in achieving its goals within the environment.

Computing the posterior p(x|y) is generally intractable and cannot be directly evaluated in most cases, especially in continuous spaces. This is usually due to the normalization term in Bayes rule p(y) also known as the marginal likelihood, which corresponds to the probability of observing y regardless of the state. For a real robot, this means knowing the unconditional probability of each measurement in advance. Under the free energy principle, a variational Bayes approach is employed to find a tractable solution. Rather than calculating an exact posterior, it is approximated through optimisation. To proceed with this approach q(x) is defined as a variational density. Then, the goal transforms into minimising a measure of the difference between p(x|y) and q(x). The Kullback-Leibler divergence (KL) does precisely this since it approaches 0 when both distributions are the same:

$$\operatorname{KL}[q(x) \parallel p(x|y)] = \int q(x) \log \frac{q(x)}{p(x|y)} dx \quad \to KL \ definition \tag{1}$$

$$= \int q(x) \log \frac{q(x)}{\frac{p(y,x)}{p(y)}} dx \quad \to chain \ rule \tag{2}$$

$$= \underbrace{\int q(x) \log \frac{q(x)}{p(y,x)} dx}_{\text{surprise}} + \underbrace{\log p(y)}_{\text{surprise}} \to \log \ product \quad (3)$$

$$= F + \log p(y) \ge 0 \tag{4}$$

$$= \mathrm{KL}[q(x) \parallel p(y, x)] + \log p(y).$$
(5)

F, in Eq. 4, is defined as the variational free energy (VFE) and measures the divergence between the variational density q(x) and the joint distribution p(y, x), as shown in Eq. 5.

The VFE can be evaluated because it depends on $q(\mathbf{x})$, our proposed model distribution, and the robots's knowledge about the environment obtained by sampling it from p(y, x) with its sensors. Notably, since the KL divergence is always positive, the VFE serves as an upper bound on the surprise: $F \geq -\log(p(y))$, which quantifies the atypicality of events through the negative log probability of sensory data. Thus, optimizing F is equivalent to evaluating the posterior density. In the ideal scenario (e.g., no noise), when the model accurately captures the real generative process, $KL[q(x) \parallel p(x|y)]$ is zero, and F becomes the marginal likelihood or surprise. The primary advantage of this formalism is that it transforms the intractable Bayesian inference problem into an optimization problem, as shown in the next section. Crucially, for AIF agents, both the state and action are inferred simultaneously by optimizing F, although we will not use the action part here.

When performing planning in discrete active inference we introduce the action plan or policy (π), where the best plan is the one that minimises the Expected Free Energy (EFE). Again this is solved through variational inference and tractable factorization of the posterior densities [10]. The policy distribution is computed as follows: $q(\pi) = \operatorname{softmax}(EFE(\pi))$.

3 State estimation in AIF

Given the above definitions, we now derive an effective procedure for estimating the internal state q that will later become the core of the robot's internal model. The goal is to find the state values that minimise the free energy F in Eq. ??, since $F \ge 0$ and reducing it reduces the difference between the model q(x) and the probability of the world being in state x given a measurement y, p(x|y).

$$x = \arg\min_{x} F(x, y) \tag{6}$$

This involves the estimation of two components: the mean estimate and the associated confidence (precision or inverse covariance) of the estimate. Under the free energy principle framework, both can be estimated using the first two gradients of the free energy.

The VFE has this closed form under the Laplace and mean-field approximations [19][4] and is defined as:

$$F(\mu, y) \approx -\ln p(\mu, y) - \frac{1}{2}\ln(2\pi\zeta^*)$$
 (7)

where μ is the estimated mean of the variational distribution q under the Laplace approximation and ζ^* is the optimal covariance of F(x, y), see [4]. We can suppress the constant optimal variance term and express $F(\mu, y)$ as

$$F(\mu, y) \approx -\ln p(\mu, y) = -\ln p(y \mid \mu)p(\mu) \tag{8}$$

which will be our target cost function to minimise. The next step is to give $p(\mu, y)$ a more concrete form. We start by defining a function,

$$y = g(\mu) + z \tag{9}$$

that generates a measurement y, i.e. a lidar sweep or a room's corner position, as a function of the estimated current state of the world μ , plus some random noise z. In the scene graph representation, it will be a geometric transformation of some object feature coordinates, i.e., a room's corner, to the sensor coordinate system, with a final covariance that is the result of transforming the original one through all the intermediate steps, see Figure 1. Solving for z and assuming it comes from a normal distribution,

$$z = y - g(\mu) = \frac{1}{\sqrt{2\pi\sigma_z}} \exp(-\frac{z^2}{2\sigma_z})$$
(10)

The likelihood $p(y \mid \mu)$ expresses how likely it is to obtain a certain measurement y given that the internal model of the world, including the robot, is at μ . Then it can be written as a random variable, z, whose mean is the estimation $g(\mu)$,

$$p(y \mid \mu) = \frac{1}{\sqrt{2\pi\sigma_z}} \exp\left(-\frac{(y - g(\mu))^2}{2\sigma_z}\right)$$
(11)

The second factor in Eq. 8, $p(\mu)$ represents the prior knowledge about the state of the world before any measurements are taken. If that initial state is defined as $\bar{\mu}$ and assuming that the elements of the world are static, the updated state will fluctuate around the prior as,

$$\mu = \bar{\mu} + w$$

The random difference $w = \mu - \bar{\mu}$ is taken as normal distribution with mean $\bar{\mu}$ and variance σ_w governing how $p(\mu)$ diverts from its mean,

$$w = p(\mu) = \frac{1}{\sqrt{2\pi\sigma_w}} \exp\left(-\frac{(\mu - \bar{\mu})^2}{2\sigma_w}\right)$$
(13)

We can apply now the properties of logarithms to $F(y, \mu)$,

$$F(y,\mu) = -\ln(p(y \mid \mu)p(\mu)) \quad \downarrow \text{ product rule}$$

= $-\ln p(y \mid \mu) - \ln p(\mu) \quad \downarrow \text{ inverse property}$
= $\frac{1}{2\sigma_z}(y - g(\mu))^2 + \frac{1}{2\sigma_w}(\mu - \bar{\mu})^2 + \frac{1}{2}\ln(\sigma_z \sigma_w),$ (14)

If our internal model represents many different objects, each one will have a different $g_k(\mu)$ prediction function. Defining $\varepsilon_{y,k} = y_k - g_k(\mu)$ as the sensory prediction error caused by an element k represented in the internal state, i.e, the pose error between a predicted 3D feature and its real measurement, and assuming that the state does not have internal dynamics and that the perceived objects are independent, F can be expressed as a quadratic cost function,

$$F(y,\mu) = \sum_{k} (\varepsilon_{y,k}^{T} \Sigma_{k}^{-1} \varepsilon_{y,k})$$
(15)

where Σ_z^{-1} represents the inverse covariance of the prediction $g_k(\mu)$ that modulates how much confidence is attributed to each term with respect to the total sum of errors.

4 DSG as generative models

The hypothesis we present in this paper states that DGS can be embedded in a control architecture and behave as a reasonable approximation to these generative models so we can use discrete active inference for perception and planning. To elaborate our argument, we define a specific kind of DSG as a graph G = (N, E) where N is set of nodes E is a set of edge, in which each node $n_i \in N$ represents a specific concept from a set \mathbf{C} : (robot, room, door, object_x) and each directed edge can denote either a probabilistic geometric transformation in SE(3), or a logical predicate in the set \mathbf{P} : (connected_to, on_top, same_as, ...) When nodes are connected by SE(3) edges, they are constrained to form a kinematic tree that represents the hierarchical spatial relations of the objects in the scene. Logical edges can break this constraint since they do not affect the geometry of the tree. Each scene represented this way can be seen as a tree of objects, including rooms, connected by springs whose stiffness is determined by how the robot's measurements match the model's elements in addition to other a priori constraints.

This definition of G encodes infinite possible instances of scene graphs, representing a wide variety of indoor spaces furnished with many different elements. See [15] for a recent but different use of space grammars in the context of scene graph construction. Rooms and doors are the basic concepts necessary to build a large-scale space using a semantic representation. Space is considered large-scale if its relevant structure is at a scale larger than the sensory horizon, so knowledge of the structure must be acquired from exploration within it [17]. The robot and everyday objects, including the doors, are elements contained inside the rooms, giving them the necessary semantic content to allow for re-identification in loop-closing and human-robot natural interaction. Based on this initial definition, we extend our DSG with the following features:

- The graph must be able to be created incrementally, starting from scratch and building on a spatial syntax of human-shared concepts. This condition forces a high level of autonomy in the robot and rejects dependency on a externally provided map. Also, the graph is created as a network of instantiated concepts, meaning that fragments of the outside space are understood as long as there are concepts, i.e. fragments of code, that can recreate it. There is no need for an underlying metric representation.
- The creation and updating of the graph must run in real-time, serving as a supporting infrastructure for the completion of more generic robot tasks. A reasonable way to achieve this is to integrate the graph in a robotics cognitive architecture and assign it the role of its working memory. This condition supports the goal of leveraging active inference implementations to more complex task oriented robot behaviours.
- The nodes in the graph are grounded instances of concepts that represent, along with the relationships among them, the ongoing belief of the robot about the world and itself. Consequently, the graph acts as an internal compact representation for the intelligent agent.
- Grounding an instance of a concept is a process that converts a measurement or detection into a belief, and a grounded belief into a predicted detection, creating a sustainable dynamics that ties the robot to the environment. In this context, grounding involves two steps. The first one begins with a new detection and proceeds to stabilise that element into a fixed belief, causing the graph to change its structure and adapt to better model the world. The second one keeps the constructed model aligned with the world by minimising all potential divergences caused by measurement errors, or by removing elements that are no longer detectable.
- All changes in the graph consist of editing operations, such as creating, deleting, and updating nodes and edges.

The cost function in Eq. 15 represents the variational free energy that must be minimised to keep the internal model aligned with the world. Given a defined state of G, that is, a specific configuration of objects and poses, the objects' parts can be projected back to the sensor spaces and compared to the real measures. Those are the $g_k(\mu)$ functions defined above. The resulting error function F can be derived with respect to the pose parameters in μ and minimised using an iterative procedure. However, the addition and removal of nodes in G is outside this mathematical framework. If an object is detected in the scene by the perception algorithms but is not represented in the DSG, no amount of smooth displacements of the existing nodes will reduce the surprise to its previous levels. The world has become more complex in ways that the model cannot smoothly compensate for. The correct action is to use the suggestions from the perception algorithm and hypothesise the existence of a new object. Then follow a procedure to validate it, fix the belief, and finally feed it into the model so that it can start making the right predictions. It is interesting to note that this discontinuous operation where one or more candidates are generated and brought into examination is very close to how abduction is defined in logic.

5 A minimal example

We describe here a simple example of a robot moving in an indoor space. Figure 1 shows a DSG in a state in which the robot, the blue node, has already detected and recognised the containing room in red, and it has also interiorised three objects in the room depicted as orange circles.

The image shows three situations of increasing complexity. The left is the discrete case, in which the space is represented as a grid and the robot and objects are placed using discrete indices. The prediction of an object y_i in the robot's frame is computed as a discrete change of coordinates.



Fig. 1: Minimal examples. Left, the robot position is discrete and estimated from the projection of the modelled objects. Center, the robot position is continuous. Right, the robot and the objects positions are estimated simultaneously.

In the middle box, the robot pose x_t with respect to its parent element (room) is defined as a continuous variable, namely an SE(3) element. The likelihood of observing an object y_t^i in the scene is still defined by a mapping $A = p(y_t^i|x_t)$,

where A is the geometric function that transforms the coordinates of the object y^i into the robot frame, by composing all intermediate transformations. In the figure, these transformations are represented by the arrows that go from the orange nodes to the springs on the bottom side. Note that the real computation involves the concatenation of the transformations in the edges that go up to the room and down to the robot. Pictured as magenta circles, the measurements of those same objects are made directly in the robot frame. Each of the small springs placed in between symbolise the matching error or, equivalently, the portion of total free energy available to the system. The goal is to minimise this free energy, as it reduces the difference between the real position of the robot (true distribution) and the estimated one (variational model). Since all other edges are frozen, all this energy has to be absorbed by the one connecting the room to the robot.

The total error is, using the expression in Eq. 15,

$$F(y,\mu) = \sum_{i=1}^{N_{obj}} (y_i - SE_i(x_i))^T \Sigma_i^{-1} (y_i - SE_i(x_i))$$
(16)

where y_i are the measured coordinates of the object in the robot's frame and $SE_y(x)$ transforms the room-defined coordinates of the object stored as part of the model into the robot frame. This transformation has to pass through the current robot position in the room frame, where the localisation error accumulates. σ_y^2 accounts for the uncertainty of the transformation.

Finally, note that we can add any type of sensor instead of the geometrical transformation if we know the data generation function that transforms the state into the observation manifold.

The third box in Figure 1 shows the case in which we want to update not only the pose of the robot but also the location of all elements in the graph, given some of their measures. This is a version of the semantic SLAM problem. Since we assume that each element has been introduced with some uncertainty in the form of a covariance matrix, the update of the positions is weighted by the inverse of this matrix. Objects that are assumed to be correctly positioned with respect to their parent elements will receive little variation. The right part of figure 1 shows the variable edges in all objects. Following the same procedure as before, the coordinates of each visible object in the graph are transformed to the robot frame and then compared with its measured coordinates. As before, each match creates a loop whose coordinates at both ends will differ due to the imprecise knowledge of the true distribution in the world. The sum of the error terms in equation 16 is modified to expose each pairwise transformation in the graph,

$$F(y,\mu) = \sum_{\langle i,j \rangle \in \mathbf{C}} (y_i - SE_{i,j}(x_j))^T \Sigma_{i,j}^{-1}(x_i - SE_{i,j}(x_j))$$
(17)

where x_i and x_j are objects related by the $SE(3)_{i,j}$ transformation with covariance $\Sigma_{i,j}$. **C** is the set of all pairwise geometric constraints encoded in the SE(3) edges. With this error function, all edges will receive a portion of the free energy in the form of a gradient-orientated vector scaled by the edge inverse covariance.

All these versions of F are optimised in our experiments with the library g2o [18]. Once the problem is defined, the algorithm returns the new poses of the robot and the objects that globally minimise the free energy.

6 An example with a simulated exploring robot

To further illustrate how the DSG can be integrated into the AIF framework, we present a case study using a simulated robot in a two-room scenario. The robot starts with no knowledge of its environment and ends with an internal model of it that has been incrementally constructed. In the experiment, we use the COR-TEX architecture [6] and benefit from its distributed memory to implement the DSG with the specific requirements described above. To run the experiment, we used a minimal configuration of the architecture with five agents. All agents can edit the DSG (working memory), and each has its own independent behaviour pattern. They can refrain from writing to the DSG and thus influence other agents depending on the graph content. The overall observable behaviour of the robot is the result of the interaction of these agents through the graph. The agents are:

- base-controller, drives the robot to a target element in the graph.
- room-detector, detects rooms from raw LiDAR data and inserts them in the graph (structural learning). Rooms are defined as parametrised orthohedra.
- door-detector, detects doors from raw LiDAR data and inserts them in the graph (structural learning)
- free-energy-optimizer, reduces the free energy in the graph, transforming it into changes in the pose of the represented objects (functional learning)
- scheduler, solves conflicting demands of resources while satisfying some high level task.

By structural learning we refer to changes to the DSG that alter its structure (additions, deletions or reorderings). Functional learning refers to changes Table 1 shows the sequence of events that follow during the experiment and affect the robot's internal model and its behaviour.

6.1 Row 1. Initial position

The robot starts without any knowledge of its environment. This situation is reflected in the first row of the table, with a robot node connected to a dummy root node. There is no need for a world reference frame.

6.2 Row 2. Room detection

The room-detector agent detects a potential room (measurement) and checks if there is already a room in the graph that can generate features that match the data just obtained (the likelihood p(y|x)). Since there is no room, the agent triggers a stabilisation process aimed at reducing the uncertainty in the estimation of the room parameters. This process first creates a room-measured node in the robot frame and proceeds by driving the robot to the estimated centre (the mean of the LiDAR points), registering all intermediate poses and detected corners, performing a batch optimisation on the trajectory, and finally accepting the most probable room size and orientation with respect to the robot. The reason for this costly initialisation process is to ensure low uncertainty when inserting objects belonging to the scene infrastructure (room, door, large furniture, etc.) so that they can be used later to constrain the poses of incoming objects. In a sense, the scene is constructed as a scaffold, with new objects leaning on older ones. The has-intention edge indicates the agent's intention to send the robot to the centre of the still measured-room. The scheduler agent must approve this request for the *base-control* agent to execute it.

6.3 Row 3. Stabilisation of the room

When the robot reaches the centre of the room, a new node is inserted into the graph as a nominal element (a part of the internal model), and the robot is connected to it to satisfy the container-containing relationship. The room is inserted with all its internal structure, as shown in the capture in Table 1. The four walls are connected to the room, each one bringing a local coordinate system that is used to locate the corners and other upcoming objects. From this point on, the room enters the grounding dynamics defined in Section 4. The *roomdetector* agent uses the nominal room in the graph to predict the coordinates of its constituent elements (corners, wall lines) back to the sensor space, where they are compared with the new measurements. The measurements are shown in the capture as nodes hanging from the robot with names *coner_x_measured*.

6.4 Free energy minimisation

Each imperfect prediction matched is a contribution to the free energy. These differences between the nominal and measured objects are detected by the *free-energy optimiser* agent, which reacts running the minimisation rule in Equation 17 for the entire graph. The actual implementation uses the g2o library [18]. Since the nominal objects will have very low uncertainty, most of the free energy is absorbed by the room-to-robot SE(3) edge, effectively re-localising the robot w.r.t the room frame.

6.5 Row 4. Door acquisition

Once the room is stabilised and under the prediction-update grounding dynamics, the *door-detector* agent activates to detect door frames (without the leaf) using an algorithm that applies the constraints imposed by the containing room, such as doors are on the walls, doors have maximum and minimum sizes, etc. Once the door is detected, the same active process, as for the room, is triggered to stabilise it with a minimum final uncertainty on its position and shape. In this case, the robot is requested through the *has_intention* edge to go to a point placed on the normal to the door plane.

6.6 Row 5. Door stabilisation

This capture shows the resulting graph after the door is stabilised. A node is added and connected to *wall_3*. Again, the door node includes its own coordinate system, so local references can be defined as robot targets. Once the door is stabilised, it also enters the prediction-update grounding dynamics. The *door detector* agent reads the internal structure of the door, that is, the start and end positions of the door in the wall frame, and transforms them to the robot frame, where they are compared with the most recent measurements. The resulting loops are detected by the *free-energy-optimiser* and included in the global minimisation of free energy. With each new nominal object added to the graph, the robot gets more ties to the room, i.e. a better knowledge of it. The situation can be visualised as elastic ropes connecting the robot to the grounded objects (to their parts actually) that tighten only when the measurements do not match the predictions.

6.7 Row 6. Door cross affordance activation

The last part of the experiment shows the robot moving into the next room. We use here the classic idea of affordances [12] to link actions with objects. In Section 7 we discuss how this approach can be related to AIF. The idea is that all concept agents (detectors) provide affordances for their grounded instances. These affordances encapsulate small plans in the form of behaviour trees that the robot can perform with the object. When an object is grounded, the agent places nodes named aff_X connected through symbolic has edges. For example, a door might have the aff_cross , open, close or knock. The scheduler agent is programmed to prioritise the exploration of new rooms, as this will entail the construction of a more complete map of the environment. It operates by selecting the best affordance according to its internal plan. The caption in row 6 of Table 1 shows the aff_cross being selected and executed until the robot crosses the door. From there, the whole series starts again, creating a new room and connecting it to the old one through the door.

This example shows how the AIF state update equations derived in section 4 can be integrated with a DSG that plays the role of a working memory in a cognitive architecture. In the simulation, the robot starts with no knowledge of the world and is able to construct an internal model that correctly predicts the stream of incoming sensory experiences. The model is also built by directly instantiating semantic concepts shared by humans, making it understandable to developers. There is no need for an underlying grid metric representation, as in [14], but it can be created at any time if required. The combination of these two approaches opens up a wide range of possibilities, on the one hand extending the core concepts of AIF to more complex real-world robotic tasks and, on the other hand, providing a formal umbrella to guide the complexity of robotic architectures into more predictable places.

7 Challenges

The hypothesis presented here and the preliminary examples that support it raise many interesting questions and challenges. In this final section, we propose some reflections on a few of them.

The scalability of hand-crafted concepts Maybe one of the most controversial features in the proposal, especially in these days of data-driven algorithms, is the use of hand-crafted detectors. There are several reasons for this. In this architecture, each concept is handled by an agent that controls all its life cycle, including the detection, stabilisation, grounding dynamics, and proposal of affordances. Detection is only part of it and could be done by DNNs, as would be the case for the initial detection of smaller household objects with the YOLO network. In addition, larger objects and the room itself could take advantage of semantic labelling networks. But in the same way as we use the vast knowledge condensed in the weights of a DNN to classify regions of the scene, we can also use basic common knowledge about rooms to introduce hard constraints and write a simple but very efficient detector. For example, the most frequent rooms in our culture are rectangular and within a certain size range. Finding line segments, right-angle corners, and ultimately rectangles in the raw LiDAR data can be done very efficiently, especially with 3D LiDAR devices that can provide readings from the upper part of the room. Another reason to make the inner parts of the object explicit is that the prediction part of the Bayesian inference (the likelihood p(y|x)) can be made opportunistic, in the sense that it can use only parts of the object, such as one corner of the room or a wall line, when the robot location prevents a complete detection. Even if the hand-crafted algorithm is quite comprehensive, there will always be edge cases in which false positives or negatives predominate. For example, L-shaped rooms, rooms separated by a wide arch, narrow hallways, or very crowded rooms. In those cases, improving requires a learning algorithm, both for a DNN and the classical solution. In the first case, new labelled examples would be required along with an appropriate retraining to ensure that there is no catastrophic forgetting. Those examples would have to be provided by someone outside of the robot, or would have to be synthetically created using an inner simulator, diffusion models [31] or a combination of both. In the second case, standard code is not derivable, so it is not possible to force a smooth improvement that extends the concept to include the outliers. There are, however, some possibilities that could be explored such as symbolic regression [21], code generation with specialized LLMs [32]. In general, the extension of a seed concept formed only by an initial geometric

description into an entity that can trigger actions in the robot, compose with other objects, expand with multimodal dimensions during the robot's lifetime, or ground generic semantic knowledge is still an open question at the core of what intelligent robots need to make progress. An initial work in this direction is [5]

Hierarchical representations in generative DSG As stated in [14], hierarchical representations are at the core of spatial perception in robots. Scene graphs naturally provide this feature that can be exploited in different interesting ways. In Section 6.2, we used the analogy with a scaffold to explain why the robot tries to be sure before admitting a new object to the model. Linking with the previous section, if the objects' internal structure is available, then inference can be made hierarchical, using the posteriors (fixed beliefs) of the stable objects as priors for the new ones. For example, knowing the geometry of the room the robot is inside, the room detector biases all LiDAR measures to those falling on the walls. Accordingly, given a door placed on one of the room's walls, searching for the door-knob would restrict all data measures to a small area defined by the prior probability of finding a knob in a door. The benefits of hierarchical representations clearly extend to long-term memories, planning, and reasoning.

Dynamic perception and planning. In the AIF formalism, planning comes as the minimisation of the *expected* variational free energy. The robot only needs to visualise its goal in the sensor space to obtain an optimal sequence of actions that will drive it to a situation where the sensors readings match the goal. In more algorithmic contexts where not all elements are derivable or analytic, this boils down to, possibly informed, computation of many sequences of imaginary actions applied to an internal model that changes when it is affected by a virtual action. An interesting way to achieve this would be to integrate a self-calibrated, internal simulator into the robotics architecture. This simulator would provide the $p(x_t|x_{t-1}, a)$ part of the generative model with x still a continuous variable and $a_i \in \mathbf{A}$, the set of robot actions. As a first approach, planning would involve detaching the DSG from the robot's body, attaching it to the synthetic sensors provided by the simulator, and trying alternative action sequences equipped with the best possible heuristics until the virtual goal state is reached. Finally, reattach the DSG to its body and execute the plan. Other solutions can be considered to avoid the freezing situation, but not without a considerable increase in the software complexity of the underlying architecture.

Learning new concepts as surprise minimisation Learning new concepts from the available internal resources and from access to external sources in response to a strong surprise, i.e. a perception anomaly, is maybe the open challenge that could bring more drastic advances to this framework. In the work presented, there are two levels of reaction to the surprise caused by the error between the predicted and the measured values. If there is an object in the internal model whose prediction is close enough, both are matched, and the free energy is distributed through the graph. If there is no nominal object providing close enough values, then a new object (concept instance) is proposed for validation. Once validated, it is inserted into the graph. But what if no concept instance is good enough to explain the anomaly? Learning new concepts that can be integrated in a running architecture in real time and that are understandable by humans is a hard open problem. On way to approach it might be combining the tools from causal reasoning [22][25], common sense ontologies [3] and automatic code generation , maybe with specialised large language models [20].

8 Acknowledgments

This work has been partially funded by the Spanish Ministry of Science and Innovation TED2021-131739-C22, supported by MCIN/AEI/10.13039/501100011033 and the European Union "NextGenerationEU"/PRTR, by the Spanish Ministry of Science and Innovation PDC2022-133597-C41 and by FEDER Project 0124 EUROAGE MAS 4 E (2021-2027 POCTEP Program).

References

- Iro Armeni et al. 3D Scene Graph: A Structure for Unified Semantics, 3D Space, and Camera. 2019. arXiv: 1910.02527 [cs.CV].
- [2] Sebastian Blumenthal et al. "A scene graph based shared 3D world model for robotic applications". In: 2013 IEEE International Conference on Robotics and Automation. 2013, pp. 453–460. DOI: 10.1109/ICRA.2013.6630614.
- [3] S. Brunner, M. Kucera, and T. Waas. "Ontologies used in robotics: A survey with an outlook for automated driving". In: 2017 IEEE International Conference on Vehicular Electronics and Safety (ICVES). IEEE, 2017, pp. 81–84.
- [4] Christopher L. Buckley et al. "The free energy principle for action and perception: A mathematical review". In: Journal of Mathematical Psychology 81 (Dec. 2017), pp. 55-79. ISSN: 0022-2496. DOI: 10.1016/j.jmp.2017.09.004. URL: https://www.sciencedirect.com/science/article/pii/S0022249617300962 (visited on 06/26/2024).
- [5] Pablo Bustos et al. "INSIGHT: The quest for causal explanations". In: WAF 2023. Aranjuez, Madrid, 2023.
- [6] Pablo Bustos García et al. "The CORTEX cognitive robotics architecture: Use cases". In: Cognitive Systems Research 55 (2019), pp. 107–123. ISSN: 1389-0417. DOI: 10.1016/j.cogsys.2019.01.003. URL: http://www. sciencedirect.com/science/article/pii/S1389041717300347.
- [7] Xiaojun Chang et al. "A Comprehensive Survey of Scene Graphs: Generation and Application". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.1 (2023), pp. 1–26. DOI: 10.1109/TPAMI.2021. 3137605.

- [8] Yun Chang et al. "Hydra-Multi: Collaborative Online Construction of 3D Scene Graphs with Multi-Robot Teams". In: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2023), pp. 10995– 11002. DOI: 10.1109/IR0S55552.2023.10341838.
- [9] Yuren Cong et al. "Spatial-Temporal Transformer for Dynamic Scene Graph Generation". In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). 2021, pp. 16352–16362. DOI: 10.1109/ICCV48922.2021. 01606.
- [10] Lancelot Da Costa et al. "Active inference on discrete state-spaces: A synthesis". In: Journal of Mathematical Psychology 99 (2020), p. 102447.
- C. Galindo et al. "Multi-hierarchical semantic maps for mobile robotics". In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2005, pp. 2278–2283. DOI: 10.1109/IROS.2005.1545511.
- [12] James J. Gibson. The Ecological Approach to Visual Perception. Boston: Houghton Mifflin, 1979.
- [13] Nathan Hughes, Yun Chang, and Luca Carlone. "Hydra: A Real-time Spatial Perception System for 3D Scene Graph Construction and Optimization". en. In: *Robotics: Science and Systems XVIII*. Robotics: Science and Systems Foundation, June 2022. ISBN: 978-0-9923747-8-5. DOI: 10.15607/ RSS.2022.XVIII.050. URL: http://www.roboticsproceedings.org/ rss18/p050.pdf (visited on 06/08/2024).
- [14] Nathan Hughes et al. "Foundations of spatial perception for robotics: Hierarchical representations and real-time systems". In: *The International Journal of Robotics Research* 0.0 (0), p. 02783649241229725. URL: https://doi.org/10.1177/02783649241229725.
- [15] Gregory Izatt and Russ Tedrake. "Scene Understanding and Distribution Modeling with Mixed-Integer Scene Parsing". In: 2021. URL: https:// api.semanticscholar.org/CorpusID:244101679.
- [16] Ue-Hwan Kim et al. "3-D Scene Graph: A Sparse and Semantic Representation of Physical Environments for Intelligent Agents". In: *IEEE Transactions on Cybernetics* 50.12 (Dec. 2020), pp. 4921–4933. ISSN: 2168-2275. DOI: 10.1109/tcyb.2019.2931042. URL: http://dx.doi.org/10.1109/ TCYB.2019.2931042.
- Benjamin Kuipers. "Modeling Spatial Knowledge." In: International Joint Conference on Artificial Intelligence (Jan. 1977). MAG ID: 2498415630 S2ID: bc34f752ef60a252a0ea547119183e1382aadc4b, pp. 292–298. DOI: 10. 1207/s15516709cog0202_3.
- [18] Rainer Kummerle et al. "G²o: A general framework for graph optimization". en. In: 2011 IEEE International Conference on Robotics and Automation. Shanghai, China: IEEE, May 2011, pp. 3607-3613. ISBN: 978-1-61284-386-5. DOI: 10.1109/ICRA.2011.5979949. URL: http://ieeexplore. ieee.org/document/5979949/ (visited on 06/15/2024).
- [19] Pablo Lanillos et al. "Active inference in robotics and artificial agents: Survey and challenges". In: arXiv preprint arXiv:2112.01871 (2021).

- [20] Michael R. Lyu et al. Automatic Programming: Large Language Models and Beyond. 2024. arXiv: 2405.02213.
- [21] Nour Makke and Sanjay Chawla. Interpretable Scientific Discovery with Symbolic Regression: A Review. 2023. arXiv: 2211.10873 [cs.LG]. URL: https://arxiv.org/abs/2211.10873.
- [22] M. Ben Messaoud, P. Leray, and N. Ben Amor. "Integrating ontological knowledge for iterative causal discovery and visualization". In: *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty.* Springer, 2009, pp. 168–179.
- [23] Z. Papp, C. Brown, and C. Bartels. "World modeling for cooperative intelligent vehicles". In: Intelligent Vehicles Symposium. 2008, pp. 1050–1055.
- [24] Thomas Parr, Giovanni Pezzulo, and Karl J Friston. Active inference: the free energy principle in mind, brain, and behavior. MIT Press, 2022.
- [25] Judea Pearl and Dana Mackenzie. The Book of Why: The New Science of Cause and Effect. New York: Basic Books, 2018. ISBN: 978-0-465-09760-9.
- [26] Corrado Pezzato et al. "Active inference and behavior trees for reactive action planning and execution in robotics". In: *IEEE Transactions on Robotics* 39.2 (2023), pp. 1050–1069.
- [27] Antoni Rosinol et al. Kimera: from SLAM to Spatial Perception with 3D Dynamic Scene Graphs. arXiv:2101.06894 [cs]. Oct. 2021. DOI: 10.48550/ arXiv.2101.06894. URL: http://arxiv.org/abs/2101.06894 (visited on 03/25/2024).
- [28] D. Shuey, D. Bailey, and T. Morrissey. "Phigs: A standard, dynamic, interactive graphics interface". In: *IEEE Computer Graphics and Applications* 6.8 (1986), pp. 50–57.
- [29] Tadahiro Taniguchi et al. "World models and predictive coding for cognitive and developmental robotics: frontiers and challenges". In: Advanced Robotics 37.13 (2023), pp. 780–806.
- [30] Shun-cheng Wu et al. "SceneGraphFusion: Incremental 3D Scene Graph Prediction from RGB-D Sequences". In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021), pp. 7511–7521. DOI: 10.1109/CVPR46437.2021.00743.
- [31] Ling Yang et al. Diffusion Models: A Comprehensive Survey of Methods and Applications. 2024. arXiv: 2209.00796 [cs.LG]. URL: https: //arXiv.org/abs/2209.00796.
- [32] Daoguang Zan et al. Large Language Models Meet NL2Code: A Survey. 2023. arXiv: 2212.09420 [cs.SE]. URL: https://arxiv.org/abs/2212. 09420.
- [33] H. Zender et al. "Conceptual spatial representations for indoor mobile robots". In: Robotics and Autonomous Systems. From Sensors to Human Spatial Concepts 56.6 (June 2008), pp. 493-502. ISSN: 0921-8890. DOI: 10. 1016/j.robot.2008.03.007. URL: https://www.sciencedirect.com/ science/article/pii/S0921889008000304 (visited on 06/08/2024).
[34] Yifeng Zhu et al. "Hierarchical Planning for Long-Horizon Manipulation with Geometric and Symbolic Scene Graphs". In: (2021). arXiv: 2012.07277, pp. 6541–6548. DOI: 10.1109/icra48506.2021.9561548.





Table 1: Structural and functional learning in a DSG for an exploring robot

Imitation Learning for vision based Autonomous Driving with Ackermann cars

 $\begin{array}{l} \mbox{Alejandro Moncalvillo}^1, \mbox{ José María Cañas}^{1[0000-0003-4179-2211]}, \mbox{ Sergio Paniego}^{1[0000-0002-8233-333X]}, \mbox{ Roberto Calvo}^{1[0000-0002-5417-6453]}, \mbox{ and Abdulla Al-Kaff}^{2[0000-0003-0212-6400]} \end{array}$

¹ RoboticsLabURJC, Universidad Rey Juan Carlos, Spain
² Universidad Carlos III de Madrid, Spain

Abstract. Modern studies have investigated the use of vision-based perception on the field of end-to-end Autonomous Driving. The objective of this paper is to demonstrate the feasibility of the vision-based endto-end (E2E) control solutions in two AD applications: Follow-Line and Follow-Lane. For this purpose we have used PilotNet, a shallow NVIDIA neural-network-based system, to control the throttle and the steering of the car. We have gathered several datasets with supervised drivers and trained with them the models using deep-learning, for both a simulated car and a real AWS DeepRacer car. The solutions have been experimentally validated in two different setups and proved successful. All the developed materials (simulations, datasets, neural network models, etc.) are open source and publicly available ³ for the research community.

Keywords: Autonomous Driving · DeepLearning · Computer Vision.

1 Introduction

In recent years we have seen how the concept of self-driving vehicles has passed from fiction into a reality that will surely improve driving in a complex environment, enhance safety on the road, and redefine what we consider a car nowadays.

Many commercial solution are already available such as Waymo, Tesla, Huawei, Nuro... and more companies will probably be added to that list in the following years. This commercial interest has fueled the artificial intelligence research field, and thanks to the usage of hardware such as GPUs and techniques as DeepLearning and Reinforcement Learning, the development speed has been impressively amplified.

These autonomous solutions can be classified into one of 5 levels of autonomy, where the automation is increased in each level. The solutions that the former companies have made fall into the second and third level of the scale, however, only from the fourth level forward can we found full self-driving vehicles with no human intervention required. There are two kind of approaches to reach full autonomy, modular and end-to-end. In the first kind, multiple sensory components

³ https://github.com/RoboticsLabURJC/2022-tfg-alejandro-moncalvillo

(GPS, Cameras, LIDAR sensors...) collect the data, which is then processed and used as inputs for an algorithm. This applications take advantage on dividing each of the tasks, however, they can lead to a complex infrastructure and require a fine tuning of the relevant information, such as performing live camera image segmentation and labeling [13].

On the other hand, end-to-end pipelines use the raw sensor data (in our case with a little of pre-processing), which means, the datasets are usually more simple (the training is easier), and can be executed at a higher frequency.

To obtain a neural network capable of such task a variety of techniques can be applied, but there are two main kinds of methods that are used in the autonomous driving field [3]. Reinforcement Learning is a type of machine learning where an agent learns to make decisions by performing actions in an environment to maximize a cumulative reward defined by an external unknown function. It is inspired by behavioral psychology and is characterized by a trial-and-error learning process. Imitation Learning is a type of supervised machine learning where the network learns by mimicking the behaviour of an external agent. In this type the network uses the matching outputs for each input in order to adjust the internal weights and parameters. This type of learning is particularly useful in scenarios where a reward function is difficult to define and we have multiple data of experiences of an expert (application or human).

This paper presents two Autonomous Driving (AD) applications solved using Imitation Learning for the development of end-to-end visual controllers: Follow-Line and Follow-Lane. The main inspiration for this work is a master's thesis[8], where both a classification and regression neural networks were trained with Imitation Learning to achieve the goal of driving a simulated F1 car in a circuit. More recent inspiring works from the same research group and approach are [10, 11,9].

2 Related work

The SOTA (State Of The Art) solutions for AD face both perception and decision making. Most of them follow a modular approach. On it the application is decomposed in several modules, each one solving one aspect, which are combined together. Typically there are perception modules, sensor fusion, localization module, global path planning, obstacle avoidance, low level control, etc..

For instance, [5] propose a hybrid architecture that combines strategy, tactical and execution modules. Some modules are traditional and other use deep learning techniques. The strategy module defines the trajectory to be followed, and the tactical decision module employs a proximal policy optimization algorithm and deep reinforcement learning.

A few years ago some end-to-end approaches [12] were proposed, consisting of a neural network directly fed with the vehicle sensors (for instance cameras) and directly connected to the vehicle actuators. One relevant end-to-end approach is the PilotNet architecture developed by NVIDIA [2], this network has about 27 million connections and 250 thousand parameters and consist of 9 layers divided in: 1 normalization layer, 5 convolutional layers, and 3 fully-connected layers. The convolutional layers are used to obtain the key characteristics of the road images and the fully-connected ones to act as a controller for the wheel steering angle.



Fig. 1. PilotNet architecture

In the Autonomous Driving domains the simulators play an important role. Some widely used simulators are Gazebo, SUMO and CARLA. They use to debug the application code, they allow massive automatic benchmarking and testing. In addition, they can be used for generating supervised datasets, both for perception (as the ground truth in the simulated world is known) and for decision making. Those datasets can be used to train Machine Learning systems. Simulators are also key factor in the Reinforcement Learning approaches for AD. They also provide framework for assessing and evaluation of candidate solutions (for instance CARLA Leaderboard).

3 Autonomous Driving with Imitation Learning

Inside Autonomous Driving many different tasks can be identified such as lane following, autoparking, lane change, route navigation between two points, roundabouts, crossings, overtaking, etc. Some of them can be faced with a reactive controller, and that controller may be developed using Imitation Learning.

3.1 Methodology

The Imitation Learning approach for programming of an end-to-end visual robot controller proposes several steps:

- 1. Collecting a supervised dataset, while an expert agent is driving the autonomous vehicle. It records the sensory input and the supervised output. It can be recorded also while a human is teleoperating the vehicle.
- 2. Preprocessing and balancing the dataset.
- 3. Choose a neural network model.
- 4. Train the network with the balanced dataset, maybe with data augmentation.
- 5. Evaluate the performance of the trained model while driving the vehicle.

Regarding the evaluation, it can be done offline and online. *Offline evaluation* measures how similar are the outputs of the model to the supervised outputs. *Online evaluation* let's the model control the robot and measures the performance of the resulting behavior.

The autonomous vehicle is a car, with an onboard color camera looking forward as the main sensor and motors to control the throttle and steering of the vehicle. As it is an end-to-end controller, the model inputs are the camera images and its outputs are the control commands to the motors.

3.2 Simulated setup

In this setup we used ROS2 [7] Humble as the robotics middleware to control and get information from the simulated car. Gazebo [6] (Gazebo-11) is the selected simulator and PyTorch the selected neural framework.

Regarding the vehicle, a simulated Formula1 car was developed. It has two versions: holonomic and Ackermann steering. The original car model obtained from the JdeRobot repository ⁴ had an holonomic type of movement (which allows rotation in the spot, common in vacuum cleaners and indoor cylindric robots). But real cars with holonomic wheels are not very common, so we decided to modify it using a Gazebol1 plugin⁵ to get a more realistic movement. The plugin is based on the Ackermann steering geometry, which is a design principle used in the steering mechanisms of vehicles to solve the issue of wheels on a vehicle needing to trace different turning angles when making a turn (see the two models side-by-side in figure 2). This is necessary because the inner wheels have a shorter turning radius compared to the outer wheels, which results in a need to steer at a more pronounced angle. A video of the Ackermann car model can be found here ⁶.

Regarding the scenarios, several circuits were developed (Figure 3) mimic some Formula1 famous circuits as Montmeló, Nurburgring or MonteCarlo. Some

 $^{^{4}\} https://github.com/JdeRobot/RoboticsInfrastructure$

 $^{^{5}\} https://docs.ros.org/en/ros2_packages/rolling/api/gazebo_plugins/generated/$

⁶ https://www.youtube.com/watch?v=PXaBTjq7vok



Fig. 2. Holonomic model used (left), Ackermann dynamics model used (right)

of them were used as training scenarios, and the datasets were collected there. Others were used as test scenarios. They all are publicly available at the JdeR-obot repository⁷.



Fig. 3. Gazebo view of the circuits

3.3 Real AWS DeepRacer setup

In this setup, ROS2 Foxy was used to control and get information from an AWS DeepRacer robot. It is a robot which has already been used to experiment and explore solutions in the autonomous diving field [1].

 $[\]overline{\ ^{7}\ https://github.com/JdeRobot/RoboticsInfrastructure}$

The main difference between the simulated car used (Figure 2 (right)) and the DeepRacer is the scale. They both have Ackermann steering drive and a ROS2 interface that makes the communication between the process and the actuators possible, which made the application migration from one to another quite straightforward.



Fig. 4. AWS DeepRacer robotic car, with Ackermann steering

The only difference between them that required some tailoring was the camera. The image obtained from the official DeepRacer camera has a fisheye distortion, that was fixed by determining the distortion matrix using the OpenCV fisheye camera model functions⁸. The distortion arrays for our real camera are:

$$K = \begin{bmatrix} 502.482 & 0 & 320.49 \\ 0 & 502.454 & 238.255 \\ 0.0 & 0.0 & 1.0 \end{bmatrix} D = \begin{bmatrix} -0.087 \\ -0.292 \\ 0.678 \\ -0.348 \end{bmatrix}$$

As the real scenario, we constructed a small circuit in the basement of a house, with red lines to bound the lane, as shown in Figure 5. It has a total of 11 meters in length and 5 curves with different radius. It seemed appropriate for the DeepRacer and it was used to create the datasets and test the final model.

4 Follow Line application

The task of the robot is to follow the red line to complete a lap on each circuit. This application was developed with a simulated holonomic car as the initial proof of concept of vision based end-to-end control with Imitation Learning. After that we also solved it with a simulated Ackermann car, which is a little bit more difficult.

⁸ https://docs.opencv.org/4.x/db/d58/group__calib3d__fisheye.html



Fig. 5. Circuit used for the Deepracer experiment.

4.1 Expert agent and supervised dataset

To generate the dataset we used basically a PID controller, implemented in Python, as the expert agent. The same type of PID was used for the holonomic and the Ackermann dynamic car models, with modified coefficients for each one. The expert controller uses a color filter(see Listing 4.1) in order to get the car situation in respect to the line, then one of the PID controllers gives the desired linear and angular speed to stay in the line, if the line is not found the car goes backwards until it is visible again(see Listing 4.2).

```
image_hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
lower_red = np.array([0, 50, 50])
upper_red = np.array([180, 255, 255])
image_mask = cv2.inRange(image_hsv, lower_red, upper_red)
points = get_relevant_pixels(image_mask)
```

Listing 4.1. Color filter used

```
if line_not_found(points) == True:
    speed, rotation = exception_case()
else:
    if is_straight_line(points):
        speed, rotation = straight_case(points)
    else:
        speed, rotation = curve_case(points)
```

Listing 4.2. After the relevant pixels are found we classify to which case they correspond to.

To obtain the data needed for the training we deployed a similar infrastructure as the one described in [8]. As told there, since almost half of the image does not provide relevant information for our case we use cropped images in the training in order to obtain better results (see Figure 6).



Fig. 6. Dataset images: raw (left) and cropped (right).

The datasets are composed of multiple images obtained from the expert driver and a single ".csv" file that contains the information about the corresponding linear and angular speed for each frame (see Figure 6).

For the holonomic model training we used a dataset composed of 52.000 images, and for the Ackermann model the dataset has around 70.000 images.

4.2 Network model and training

The network model used in all of the cases was a variant the PilotNet [2], as it has proved good results in the literature for end-to-end learning. In our case the network structure has been slightly modified (see figure 1) to inference not only the steering angle but also the throttle, similar to the TinyPilotNet [4] one. It was implemented in PyTorch⁹, and trained using the Adam algorithm¹⁰ with a loss value obtained from the mean squared error ¹¹ between the inference values and the real ones.

To enhance the network performance we used a data augmentation technique in training to increase the information of each dataset, the one selected was the Gaussian noise injection to images.

4.3 Experimental validation and evaluation

In the offline evaluation the trained model for the holonomic model presented more resemblance to the expert agent controller than the Ackermann one (see figures 7, 8). Nevertheless, they both are able to mimic the PID controller up to a degree that enables the circuit's completion.

A typical execution can be seen at this video¹² for the Ackermann car.

One of the lessons learnt with these experimental validation is that the Ackermann car is more nervous than the holonomic one, and so more difficult to control properly. This way a successful model required larger datasets.

⁹ https://pytorch.org/

¹⁰ https://pytorch.org/docs/stable/generated/torch.optim.Adam.html

 $^{^{11}\} https://pytorch.org/docs/stable/generated/torch.nn.MSELoss.html$

¹² https://youtu.be/eaSRJ3pmuT0



Fig. 7. Frame-to-frame comparison between PilotNet output and Expert Controller output for the Holonomic car model for the follow line task



Fig. 8. Frame-to-frame comparison between PilotNet output and Expert Controller output for the Ackermann car model for the follow line task

5 Follow Lane application

After getting successful results in the Follow Line case we continued onto a case closer to a real scenario, Follow Lane. This application was solved with a simulated holonomic car and two Ackermann cars, one in the simulated setup and another in the real setup with the AWS DeepRacer. After testing the neural network architecture on the simulations we started to work on the physical world experiment.

5.1 Expert agent and supervised dataset

In the follow lane additional problems are present due to having only half of the road to maneuver when the car tries to take a curve. This issue impacts on the control of the car, mainly on the linear speed, it is necessary to have a lower linear speed when taking a curve to not get out of the road. After modifying the controllers (see Section 4.1) to complete this task we obtained results similar to the previous ones, where the model trained with the dataset obtained from the holonomic controller gives and inference value closer to the real one (see figures 9, 10).



Fig. 9. Frame-to-frame comparison between PilotNet output and Expert Controller output for the simulated holonomic car model for the Follow Lane task



Fig. 10. Frame-to-frame comparison between PilotNet output and Expert Controller output for the simulated Ackermann car model for the Follow Lane task

In this case, for the simulated holonomic model training we used a dataset composed of 40.000 images, and for the simulated Ackermann model the dataset has around 67.000 images.

We simplified the controller (see Section 4.1) to only have two types of output for the throttle, positive or negative, leaving the steering as a proportionate control but only taking values with one decimal digit (see Listing 5.1). Additionally to the camera problems described in section 3.3, we encountered that the brightness was not always the same and had to be taken into consideration when obtaining the dataset (see figure 11).

```
if line_not_found(points) == True:
    speed, rotation = exception_case()
else:
    speed, rotation = normal_case(points)
```

Listing 5.1. Control statements for AWS deepracer



Fig. 11. Two images taken from almost the same position on two different days at different times.

For the training we used the same policy as in the other cases, providing a cropped version of the image as input values (see Figure 12), the dataset used has around 20.000 images.



Fig. 12. Cropped image used for training

5.2 Network model and training

The network model architecture, training and the data augmentation follow the same methodology as the one described in section 4.2.

5.3 Experimental validation and evaluation

After balancing the datasets and training the model we obtained quite good results for this circuit and can be seen in this video¹³ (see Figures 13,14 for reference).

The offline evaluation comparing inference outputs with the supervised outputs see Figure 15.

¹³ https://www.youtube.com/watch?v=M_F-TUWN21k



Fig. 13. Images from Ackermann dynamics car follow lane neural network application execution $% \left[{{\mathbf{F}_{\mathrm{s}}}^{\mathrm{T}}} \right]$



Fig. 14. Images from DeepRacer car follow lane neural network application execution



Fig. 15. Frame-to-frame comparison between PilotNet output and Expert Controller output for the DeepRacer

A typical execution can be seen at this video¹⁴ for the Ackermann vehicle. The model successfully learned to go forward keeping the vehicle inside the lane and even to go back sometimes, when the car slightly goes out of the lane, as

¹⁴ https://www.youtube.com/watch?v=MA6YnOSDr_8

the expert agent did and was recorded in the training dataset. In addition, the trained model probed to be robust to certain changes in lightning conditions.

6 Conclusions

This paper has showed that Imitation Learning is a possible approach for the "programming" of autonomous driving robots. A simple Follow Line application and a more useful Follow Lane application have been solved with it. Both are vision based reactive applications. The first was the proof of concept using the Gazebo simulator and an holonomic car model. The second is closer to real life, as it uses vehicles with Ackermann dynamics and real small car.

Following Imitation Learning methodology, an expert agent or even a teleoperated execution were used to generate large datasets. The wider and heterogeneous the dataset, the better. Then a neural model was trained with it. We selected the shallow PilotNet network architecture, which is capable of achieving multiple objectives related to autonomous driving.

The inferences of that trained neural model were used to actually drive the vehicle, connecting the onboard camera to the network input and the network output to car actuators. Both applications have been experimentally validated, in both simulated (using Gazebo simulator) and real settings (using the AWS DeepRacer robot).

As future lines, we plan to use commercial cars, both in CARLA simulator and real ones, and to explore TransferLearning with them. Also we want to extend the scope of the FollowLane application to also show traffic adaptation to other vehicles in front. In addition we have also started to use Imitation Learning to aerial robots, drones, both simulated and real ones.

Acknowledgments. This work is supported by (GAIA) Gestión integral para la prevención, extinción y reforestación debido a incendios forestales, Proyectos de I+D en líneas estratégicas en colaboración entre organismos de investigación y difusión de conocimientos TRANSMISIONES 2023. Ref PLEC2023-010303 (2024-2026) by Agencia Estatal de Investigación de España.

References

- Balaji, B., Mallya, S., Genc, S., Gupta, S., Dirac, L., Khare, V., Roy, G., Sun, T., Tao, Y., Townsend, B., Calleja, E., Muralidhara, S., Karuppasamy, D.: Deepracer: Autonomous racing platform for experimentation with sim2real reinforcement learning. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 2746–2754 (2020). https://doi.org/10.1109/ICRA40945.2020.9197465
- Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., et al.: End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316 (2016)
- Chen, L., Wu, P., Chitta, K., Jaeger, B., Geiger, A., Li, H.: End-to-end autonomous driving: Challenges and frontiers. arXiv preprint arXiv:2306.16927 (2023)

- del Egio, J., Bergasa, L.M., Romera, E., Gómez Huélamo, C., Araluce, J., Barea, R.: Self-driving a car in simulation through a cnn. In: Advances in Physical Agents: Proceedings of the 19th International Workshop of Physical Agents (WAF 2018), November 22-23, 2018, Madrid, Spain. pp. 31–43. Springer (2019)
- Gutiérrez-Moreno, R., Barea, R., López-Guillén, E., Arango, F., Abdeselam, N., Bergasa, L.M.: Hybrid decision making for autonomous driving in complex urban scenarios. In: 2023 IEEE Intelligent Vehicles Symposium (IV). pp. 1–7 (2023). https://doi.org/10.1109/IV55152.2023.10186666
- Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566). vol. 3, pp. 2149–2154 vol.3 (2004). https://doi.org/10.1109/IROS.2004.1389727
- Macenski, S., Foote, T., Gerkey, B., Lalancette, C., Woodall, W.: Robot operating system 2: Design, architecture, and uses in the wild. Science Robotics 7(66), eabm6074 (2022). https://doi.org/10.1126/scirobotics.abm6074, https://www.science.org/doi/abs/10.1126/scirobotics.abm6074
- Martínez, V.F.: Conducción autónoma de un vehículo en simulador mediante aprendizaje extremo a extremo basado en visión. Master's thesis, Máster Universitario Visión Artificial, Universidad Rey Juan Carlos (2019)
- Paniego, S., Calvo-Palomino, R., Cañas, J.: Behavior metrics: An open-source assessment tool for autonomous driving tasks. SoftwareX 26, 101702 (2024)
- Paniego, S., Paliwal, N., Cañas, J.: Model optimization in deep learning based robot control for autonomous driving. IEEE Robotics and Automation Letters 9(1), 715–722 (2023)
- Paniego, S., Shinohara, E., Cañas, J.: Autonomous driving in traffic with end-toend vision-based deep learning. Neurocomputing p. 127874 (2024)
- Tampuu, A., Matiisen, T., Semikin, M., Fishman, D., Muhammad, N.: A survey of end-to-end driving: Architectures and training methods. IEEE Transactions on Neural Networks and Learning Systems 33(4), 1364–1384 (2020)
- Wu, D., Liao, M.W., Zhang, W.T., Wang, X.G., Bai, X., Cheng, W.Q., Liu, W.Y.: Yolop: You only look once for panoptic driving perception. Machine Intelligence Research 19(6), 550–562 (2022)

Multimodal Emotional Recognition for Human-Robot Interaction in Social Robotics

Sergio García-Muñoz¹, Francisco Gomez-Donoso¹[0000-0002-7830-2661]</sup>, and Miguel Cazorla¹[0000-0001-6805-3633]

University Institute for Computing Research. Universidad de Alicante, 03690, Alicante, Spain. sgm171@alu.ua.es, {fgomez, miguel.cazorla}@ua.com

Abstract. This study explores the enhancement of human-robot interaction (HRI) through multimodal emotional recognition within social robotics, using the humanoid robot Pepper as a testbed. Despite the advanced interactive capabilities of robots like Pepper, their ability to accurately interpret and respond to human emotions remains limited. This paper addresses these limitations by integrating visual, auditory, and textual analyses to improve emotion recognition accuracy and contextual understanding. By leveraging multimodal data, the study aims to facilitate more natural and effective interactions between humans and robots, particularly in assistive, educational, and healthcare settings. The methods employed include convolutional neural networks for visual emotion detection, audio processing techniques for auditory emotion analysis, and natural language processing for text-based sentiment analysis. The results demonstrate that the multimodal approach significantly enhances the robot's interactive and empathetic capabilities. This paper discusses the specific improvements observed, the challenges encountered, and potential future directions for research in multimodal emotional recognition in HRI.

Keywords: Human-robot interaction \cdot multimodal emotion recognition \cdot social robotics \cdot humanoid robots \cdot natural language processing

1 Introduction

At the intersection of technology and personal assistance, humanoid robots represent a revolutionary field with the potential to significantly transform humanrobot interaction (HRI). In particular, the ability of these robots to understand and respond to human emotions is crucial to their effectiveness in assistive, educational, and healthcare settings. Pepper, a humanoid robot designed by SoftBank Robotics, exemplifies the application of these technologies in creating more intuitive and empathetic machines. However, despite its advanced interaction capabilities, Pepper's ability to correctly interpret human emotions remains a significant challenge.

This work focuses on enhancing Pepper's interactive capabilities through multimodal emotion recognition, integrating image, audio, and text analysis to enable more accurate and contextual interpretation of human emotions. Research into multimodal systems for emotional recognition not only expands the understanding of the user's emotional state but also facilitates more natural and appropriate responses from the robot.

Improving emotional recognition in robots like Pepper has profound implications for the future of assistive robotics. By enhancing the ability of these robots to interact in a more human-like manner, their utility can be significantly increased in scenarios where emotional support and understanding are essential, such as in caring for the elderly, educating children, or supporting people with disabilities.

This paper presents a detailed examination of the methods and technologies used to implement multimodal emotion recognition in Pepper. It highlights specific improvements in emotional recognition accuracy and discusses how these enhancements contribute to more seamless and meaningful interactions in various care contexts. Additionally, the technical challenges encountered and the solutions developed are addressed, providing a solid foundation for future research in the field of HRI.

The rest of the paper is structured as follows. First, a review of related works is presented in Section 2. Then, Section 3 describes the system and its components. In Section 4, we detail the experiments conducted to validate our approach. Finally, conclusions and future work are discussed in Section 5.

2 Related Works

The research in this papers intersects with different topics of the state-of-the-art. In the following Sections, a brief summary of relevant related work is given in order to set the background required.

On one hand, the field of assistive robotics has witnessed considerable growth, particularly with developments in humanoid robots like Pepper that can perform both assistive tasks and act as social companions. Research in this area has evolved from basic task automation to more complex interactions involving emotional and social intelligence, fundamentally aimed at improving the quality of life for humans, especially those with special needs, the elderly, or children in educational settings [10]. An example of recent work in assistive robotics is the development of Pepper, designed not only as an assistant for daily tasks but also as a social companion [17, 2].

On the other hand, human-robot interaction has expanded significantly with advances in AI and robotics, increasingly focusing on the social capabilities of robots. The concept of the "Uncanny Valley," introduced by Mori in the 1970s, continues to influence robot design, emphasizing a balance between robot humanlikeness and user comfort [5]. Current studies in HRI strive to enhance robot understanding of human behaviors and emotions to facilitate more natural and intuitive interactions.

Regarding neural networks, they play a pivotal role in enabling robots to process and interpret vast amounts of sensory data. Artificial Neural Networks (ANNs) and their subsets, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), are crucial for tasks ranging from visual recognition to processing sequential data like speech and text [13]. These technologies form the backbone of most modern robotic systems, allowing for advancements in machine perception and decision-making capabilities.

In the last years, transformer architectures have revolutionized NLP, providing a mechanism for models to weigh the relevance of different words in a sentence, regardless of their position. This approach has significantly improved the performance of language models on tasks such as translation, sentiment analysis, and text generation [15]. In robotics, transformers are increasingly used to improve robots' language understanding and interaction capabilities, enabling more sophisticated dialog systems that can engage in meaningful conversations with users [12].

Finally, the integration of multiple modalities, such as visual, auditory, and textual data, to recognize human emotions represents a significant advancement in making robots more empathetic and responsive. This approach leverages the strengths of each modality to overcome the limitations of individual data types, enhancing the robot's ability to understand and react to human emotional states in a comprehensive and nuanced manner [7].

3 Methodology

The proposed system for multimodal control of the Pepper robot involves an intricate architecture designed to facilitate various interactive functionalities. This Section is split into two primary sections: an overview of the general system structure and a detailed explanation of each module.

Starting with the general pipeline, the interaction process begins with a user recording a voice message via a mobile interface. Concurrently, a camera monitors and records facial expressions during the recording. Post-recording, emotion recognition methods process the audio, transcribed text, and captured images to determine the predominant emotion. The transcribed text undergoes intention analysis to deduce the requested action. Depending on the user's request, the system can schedule reminders, control Pepper's movement, initiate a Chat-Bot interaction, or activate object recognition functionality. Figure 1 depicts the structure of the system.

3.1 Mobile Interface Development

The mobile interface development involves two main components: a mobile application developed with React Native and a backend server running on Node.js. These components work together to facilitate the recording and transmission of voice messages from the mobile device to the central processing system.

The interface is created using React Native, allowing for the development of an intuitive and user-friendly application that can record audio. The application



Fig. 1. Interaction Flow.

includes a recording button which sends the audio file to the server for processing. The design prioritizes minimalism and visual appeal, hence it only features just a recording button.

The audio server, developed in Node.js and utilizing the Multer library for file handling, is responsible for receiving and processing audio files sent from the mobile interface. Upon receiving an audio file, the server temporarily stores it.

The conversion and storage of the received audio are managed with precise specifications to ensure compatibility and the necessary quality for further analysis. These specifications include:

- File Format: The audio is converted to WAV format, which is widely used and compatible with most audio analysis tools.
- Audio Codec: The pcm_s16le codec is used, ensuring lossless and high-fidelity encoding.
- Sampling Rate: The audio is set to a sampling rate of 16000 Hz.
- Audio Bitrate: A bitrate of 256 kbps is set, ensuring superior audio quality for detailed analysis.
- Audio Channels: The audio is processed in mono (1 channel), which is sufficient for voice analysis and reduces the file size.

The choice of these characteristics is not arbitrary; it is based on the need for the processed audio to match the original recording characteristics of the mobile device. This consistency in the recorded and processed audio characteristics allows for more accurate comparison and analysis.

3.2 Intention Recognition

The intention recognition system interprets user requests to determine the appropriate action. Utilizing labeled datasets, the model is trained on common phrases associated with specific actions. The implementation employs TensorFlow and Keras with an RNN that processes user phrases. Each phrase is tokenized, converted into numerical sequences, and transformed into feature vectors via an embedding layer. These vectors pass through RNN layers designed to recognize and remember word sequences and contexts, facilitating precise intention classification. This ensures natural interaction, improving the user experience with Pepper. Further details about the network and datasets used for training are provided in Section 4.1.

Examples of phrases corresponding to different user requests include:

- Movement: "Move forward", "Turn left."
- Object detection: "What is around me?", "Can you see the chair?"
- Reminders: "Remind me to call Maria in 5 minutes.", "Set a reminder for my meeting."
- Starting a conversation: "Pepper, I want to start a conversation with you.", "Activate ChatBot mode."

As it can be observed, certain keywords like 'move', 'turn', 'remind', 'object', or 'conversation' are repeated depending on the desired action and help the system classify intentions effectively. This recognition of linguistic patterns allows the model to process natural phrases and deduce the underlying intention without requiring a strict command format. The network has been trained to associate these phrases with the appropriate intention category.

3.3 Multimodal Emotion Recognition

This system focuses on detecting emotions through visual, auditory, and textual inputs. The system utilizes various models to classify four primary emotions: angry, happy, neutral, and sad.

Visual Emotion Detection The visual emotion detection employs a 2D Convolutional Neural Network (CNN) to analyze grayscale frames resized to a specific scale. The CNN architecture includes convolutional layers, pooling layers, and dense layers for feature extraction and emotion classification. Throughout the recording, the system continuously captures the emotion from the current frame. When the duration of the audio is determined, the system averages the emotions of the frames that correspond to the duration of the audio, ensuring a comprehensive emotional analysis. The 2D CNN model used is trained on a combination of publicly available datasets (RAVDESS, CREMA-D, Christian Mejía) with data augmentation techniques applied, followed by fine-tuning with personalized datasets. References to these datasets are provided in Section 4.2.

Auditory Emotion Detection For auditory emotion detection, features such as MFCCs, Mel-Spectrogram, Chroma, and Spectral Contrast are extracted from the audio using the *librosa* library. A 1D CNN processes these features to classify emotions based on audio signals. Data augmentation techniques, such as speed

variation and background noise introduction, enhance model robustness. The 1D CNN model is trained on combined datasets (RAVDESS, SAVEE, AESDD) with augmentation and subsequently fine-tuned using personalized data. References to these datasets are provided in Section 4.2.

Textual Emotion Detection Textual emotion detection involves transcribing audio to text, translating it into English, and employing a pretrained model to classify the text into emotions like happiness, sadness, and anger. This process ensures accurate emotion recognition from the textual content of the audio. The translation model used is the MarianMT model from the Helsinki-NLP group (Helsinki-NLP/opus-mt-es-en) [14], and the emotion classification is performed using a pretrained transformer-based model fine-tuned for sentiment analysis.

This system focuses on detecting emotions through visual, auditory, and textual inputs. The system utilizes various models to classify four primary emotions: angry, happy, neutral, and sad.

3.4 Scene Description

LLaVA, an advanced computer vision tool, enriches ChatBot interactions by analyzing the user's visual environment [8]. It identifies key details of the scene, such as location, age, and gender. The integration involves models trained to generate descriptions and responses based on visual content, enhancing the ChatBot's contextual understanding.

The responses generated by LLaVA are essential for setting the context in conversations, allowing the ChatBot to interact more effectively by understanding the user's environment and situation. For instance, the ChatBot's response to a query will vary significantly based on the environmental context provided by LLaVA. If a user seeks advice on a problem, the ChatBot's response would differ if the user is in a park versus in a clinic with a doctor. In a park, the ChatBot might provide a more casual and general advice, whereas in a clinic, the advice could be more serious and medically oriented. This context-aware interaction ensures that the ChatBot's responses are relevant and appropriate to the user's current situation, thereby improving the overall user experience.

Some examples illustrating how LLaVA analyzes different scenes and provides information that the ChatBot uses to personalize its responses are shown in Figures 2 and 3.



Fig. 2. Student in a classroom.



Fig. 3. Person in an urban environment.

Scene description module responses for Figure 2:

- Environment: Classroom.
- Gender: Female.
- Age: Child.

Scene description module responses for Figure 3:

- Environment: The person is on a street, surrounded by buildings and vehicles in the background.
- Gender: Male.
- Age: Young, probably in adolescence or early adulthood.

3.5 ChatBot Setup

The ChatBot employs the Mixtral8x7b model, an advanced open-source language model [4]. The ChatBot adapts to various interaction contexts, using inputs like the expected role, environmental information from LLaVA, user emotion, and user message. The model processes this contextualized input to generate coherent and appropriate responses. The implementation uses a RESTful API for communication between the local computer and the server hosting the models and GPUs, ensuring seamless interaction and response generation.

3.6 Object Recognition

Object recognition allows Pepper to identify and classify objects in real-time. The system captures images of the environment using a camera and processes them with the YOLOv8 model [6]. A continuous cycle captures and stores frames, which YOLOv8 analyzes to detect and identify objects based on confidence thresholds. This functionality operates in parallel with other system tasks, enabling real-time object detection and user interaction. It is important to note

that the object recognition module operates independently from the emotion recognition system, focusing solely on environmental awareness and interaction capabilities.

3.7 Alarm Creation

The alarm creation module enables users to set reminders through voice commands. The system interprets the time and message of the reminder, differentiating between units like seconds, minutes, and hours. It schedules reminders accordingly, using a multithreaded approach to manage multiple alarms and ensure timely activation. When the time is up, Pepper announces the reminder message.

3.8 Social Robot Movement

As mentioned earlier, we decided to implement our system on a commercial social robot. In this case, the robot of choice was Pepper. The movement module encompasses two main capabilities: person tracking and executing specific movements.

Person Tracking Pepper has the ability to follow a person by adjusting its position and orientation in real-time using facial detection algorithms. This tracking is based on the three-dimensional position [x, y, z] of the detected target relative to Pepper's torso reference frame, allowing precise calculation of the distance and relative orientation of the user to the robot.

This process involves constant calculations regarding Pepper's position and orientation relative to the user, based on the detected three-dimensional position. The formulas for adjusting translation and rotation speeds take these three-dimensional measurements into account to ensure consistent and continuous interaction:

- To adjust distance:

$$vel \ traslation = \pm 0.1 + (position \ X - threshold) \times 2 \tag{1}$$

- To adjust orientation:

$$vel_rotation = \pm 0.1 + |position_Y - threshold|$$
 (2)

where threshold represents the optimal distance Pepper tries to maintain from the user, and the position [x, y, z] indicates the user's location relative to Pepper's torso.

Executing Specific Movements In addition to tracking, Pepper can perform precise movements in response to detailed verbal instructions.

The process involves several key steps:

1. Transcription: The user's spoken command is converted into text using speech recognition technology.

2. Command Parsing: The transcribed text is broken down into individual words or tokens. Each token is analyzed to determine its part of speech (e.g., noun, verb, adjective), and the syntactic structure of the command is analyzed to understand the relationships between words. This helps in identifying the main action (e.g., move, rotate) and its parameters (e.g., direction, distance, angle).

Based on the parsed information, the system extracts the required action. For instance, if the command includes words like "move" or "displace," the system understands that a movement action is required. Similarly, words like "turn" or "rotate" indicate a rotation action. The system then identifies additional parameters such as the direction (e.g., forward, backward, left, right) and the magnitude of the movement (e.g., distance in meters or angle in degrees). For example, the phrase "two meters" would be parsed to set the distance parameter.

Once the command is fully interpreted, it is translated into executable instructions for Pepper. For example, if the user says "Pepper, move forward two meters," the system will understand that Pepper needs to move forward by two meters. It will then send the appropriate commands to Pepper's motion control system to execute the movement.

This robust interpretation mechanism ensures that Pepper can accurately understand and respond to a wide range of movement commands, enhancing its capability to interact seamlessly with users.

4 Experiments

This section presents the results obtained from various tests and training sessions. In all tests, the accuracy plot, the loss plot.

4.1 Intention Recognition

This section presents the results of training the RNN associated with intention recognition. The model is evaluated to determine its reliability and accuracy in detecting user intentions, a crucial aspect to ensure smooth and effective interaction between the robot and the user.

Dataset Description The dataset used for training was created manually and consists of the following classes: tracking, reminder, ChatBot, movement, and object detection. Each class contains between 70 and 90 phrases. The dataset was divided into 75% for training and 25% for testing.

Input and Output The input to the RNN consists of the tokenized phrases, which are converted into numerical sequences. These sequences are then transformed into feature vectors through an embedding layer. The output of the RNN is a classification of the phrase into one of the five intention classes.

A video demonstrating the intention recognition system can be seen in 1 .

Strengths and Limitations The model performs well when the phrases closely resemble those in the dataset, as the RNN effectively associates keywords with the corresponding class. However, there are limitations. The model cannot handle combined module requests (e.g., "Tell me what objects are in the room and then remind me to..."). This limitation arises because the RNN is trained to classify single-intention phrases and does not support the recognition of multiple intentions within a single input.

4.2 Emotion Recognition

This section presents the experiments carried out to develop the multimodal emotion recognition module. It is important to note that, as mentioned earlier, emotion in text is detected using a pretrained detector, so no experiments have been conducted regarding emotion detection from text. A video demonstrating the multimodal emotion detection system can be seen in 2 .

Datasets For the analysis and recognition of emotions in both audio and frames, carefully selected datasets known for their diversity and data quality in the field of artificial intelligence were used.

RAVDESS The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) contains audiovisual recordings of actors performing emotional expressions in a controlled manner [9]. For frame-based emotion recognition, the visual components of this dataset are used, which include facial expressions corresponding to emotions such as joy, sadness, anger, and surprise. In the context of audio, RAVDESS offers high-quality voice recordings where actors express emotions clearly and distinctly, making it an optimal tool for developing systems capable of identifying human emotions through speech.

Crema-D The CREMA-D (Crowd-sourced Emotional Multimodal Actors Dataset) provides audio and video recordings of actors expressing a variety of emotions through dialogues. Similar to RAVDESS but with its own uniqueness in the diversity of situations and emotions, CREMA-D stands out for including a broader spectrum of emotions and a wide variety of actors. This dataset is particularly useful for training models that need to recognize emotions in dynamic situations and with variations in emotional intensity [1].

¹ https://youtu.be/ILoHKfFA4CY]

² https://youtu.be/sWtVKhZLLXs

Christian Mejía The dataset created by [11] is a collection of images representing emotions, mostly sourced from the FER2013 and AFF-WILD datasets. Additionally, it also includes images generated by Artificial Intelligence.

Surrey Audio-Visual Expressed Emotion (SAVEE) The SAVEE dataset was created by researchers at the University of Surrey and consists exclusively of highquality voice recordings from native English-speaking men, making it a valuable resource but with some limitations in terms of gender representation. The dataset includes recordings from four men, graduate students, and researchers, aged 27 to 31. The recorded emotions span seven psychologically defined categories: anger, disgust, fear, happiness, sadness, surprise, and a neutral category [3].

Acted Emotional Speech Dynamic Database (AESDD) The Acted Emotional Speech Dynamic Database (AESDD) is designed for speech emotion recognition (SER), categorizing emotions into five types: anger, disgust, fear, happiness, and sadness. This dataset is used for audio-based emotion recognition [16].

Facial Emotion Detection In this subsection, the performance of the trained models for facial emotion detection will be analyzed using different neural network architectures and various datasets. The results of these trainings will allow evaluating the accuracy and effectiveness of each model in classifying emotions.

Various experiments were conducted with different architectures and datasets to determine the best approach for facial emotion detection:

- CLIP (Contrastive Language-Image Pre-training): CLIP, developed by OpenAI, was evaluated for its ability to understand and classify emotions in images. The experiments involved the general pre-trained model being evaluated on the frame datasets (RAVDESS, CREMA-D, and Christian Mejía). In all datasets, the model showed good accuracy for the "angry" and "happy" classes (between 70% and 90%, depending on the dataset), but for the "neutral" class, the highest accuracy was 30%, possibly due to the model not being trained with "neutral" text labels. The "sad" class accuracy ranged between 50% and 60%, depending on the evaluated dataset.

Four predictive models were created: the pre-trained model and three finetuned models with each dataset, including a combined dataset excluding the dataset being evaluated. Thus, three evaluation tests were conducted, one for each dataset. The best model was the one fine-tuned with the Christian Mejía dataset, achieving a 75% average accuracy across the four emotions on the RAVDESS dataset and 47% on the CREMA-D dataset (the best performance among the four models).

The pre-trained CLIP model can correctly associate images with well-described emotions (e.g., a person crying for "sad"), but it fails with images expressing no clear emotion. Fine-tuning with images from different datasets and introducing the text label "A person expressing a neutral emotion" for neutral images aimed to resolve this issue. Evaluation involved inputting four text labels ("A person expressing an emotion of ...") for each image, and the highest predicted percentage determined the classification.

- Visual Transformers: Experiments were conducted using Visual Transformers (ViT), which capture complex relationships within images efficiently. These models were tested on the RAVDESS dataset due to hardware limitations, and their performance was compared to other architectures. Despite the limitations, the ViT model achieved a 95% accuracy on the validation set, similar to the 2D CNN on the same dataset. Future work would benefit from more GPUs to evaluate the system using all datasets combined, potentially improving the robustness and generalization of the model.
- 3D Convolutional Neural Networks (CNNs): This approach analyzed entire video sequences rather than individual frames. The RAVDESS dataset, which provides video sequences, was used for these experiments. The objective was to determine whether analyzing sequences could provide better emotion detection compared to frame-by-frame analysis. However, the 3D CNN model only achieved 70% accuracy on the validation set, significantly lower than the 95% accuracy obtained by the 2D CNN and Visual Transformers on the same dataset. Due to this lower performance, the 3D CNN architecture was discarded.
- 2D Convolutional Neural Networks (CNNs): Initially, separate 2D CNNs were trained for each dataset (Crema-D, Ravdess, Christian-Mejía). Datasets without pre-segmented training and validation folders were manually segmented, allocating 75% of the data to training and 25% to validation. Subsequently, an additional test was performed applying Data Augmentation techniques to the training set to evaluate the improvement in the model's generalization capability.

The aim of these trainings was to test the performance of each model with different datasets, identify which emotion classes are best detected, and those that present the most challenges.

After extensive testing, the 2D CNN model proved to be the most effective. The final experiment combined all datasets and applied fine-tuning with a personalized dataset.

Although the fine-tuning process is performed for a single person, it does not always achieve an accuracy value close to 100%, and this has a simple explanation. From each message that the user recorded, five frames per second were extracted. This implies that even if the person is angry during the conversation, not all frames will necessarily show micro-expressions of anger; some might show neutrality. In practice, this does not pose a problem since an average weighting of all frames throughout the conversation is performed. For example, if in a seven-second conversation, 76% anger and 24% neutrality are detected, it will be assumed that the person was angry while conveying that message.

Auditory Emotion Detection In this subsection, the performance of the auditory emotion detection model using Convolutional Neural Networks (1D CNN) will be analyzed. The goal is to evaluate the model's effectiveness in predicting emotions from audio data using accuracy and loss plots and identifying which emotions are easier or harder to predict. Additionally, the performance of different datasets separately and combined will be evaluated, both with and without the application of data augmentation techniques. This evaluation will determine which data configuration provides the best performance and on which the finetuning process will be performed.

The final experiment combines all datasets with data augmentation and finetuning using a personalized dataset.

These results demonstrate significant success in fine-tuning, achieving an accuracy of 100%. It is important to note that the quality of fine-tuning significantly depends on how the user differentiates the messages of different emotions. The goal is to allow any person to create their personalized dataset in a short time, approximately 5 minutes. However, with so few data, the system might not be robust to variations such as microphone distance—for example, it would not work the same if the data were collected with the microphone 20 cm away and then predicted with the microphone one meter away—or changes in the person's tone of voice on different days.

Ideally, the system would benefit if, over a week, the user recorded four messages daily, one for each emotion, slightly varying the tone of voice and the microphone distance. Despite these limitations, the system works quite well if there are no significant changes in tone of voice and the microphone is kept at an approximate and constant distance.

A video with an interaction between an user and the robot can be seen in 3 .

5 Conclusions and Future Work

This study has successfully integrated an advanced conversational system into the Pepper robot, adapting it to user emotions through multimodal analysis of vision, audio, and text. The system has demonstrated effective operation even with moderately quality audio recording devices and in a short time for personalized adaptation through the use of fine-tuning techniques.

The implementation has transformed Pepper into a versatile agent that not only assists in simple tasks but also recognizes objects, moves in the environment, remembers events, and holds contextualized and emotionally coherent conversations. This represents a significant advancement towards humanizing robot-user interactions, providing a more natural and enriching experience.

However, challenges were encountered, such as the response speed of Mixtral, technical issues with connectivity, and the somewhat outdated hardware of Pepper. These manageable issues highlight important areas for future improvements. Regarding future work, several directions for further research and development are proposed. One key direction is the integration of real-time mapping capabilities, adopting a robot with advanced mapping technology to not only detect but also localize objects in real-time, thereby enhancing interaction and system functionality. Additionally, the implementation of physical manipulation capabilities is crucial; using a robot with fully articulated arms will allow the robot

³ https://www.youtube.com/watch?v=dkuLXqDm8v8

to transport objects at the user's request, increasing the autonomy and utility of the assistive system. Continuous updates of vision and language models, such as LLaVA and Mixtral, are necessary to keep the system effective and accurate as the state of the art in computer vision and natural language processing advances. Finally, developing a larger and more diverse dataset for intention recognition will improve the accuracy in identifying user requests, making the system more reliable and versatile in different contexts.

In conclusion, integrating multimodal emotion recognition technologies into humanoid robots like Pepper holds tremendous potential for transforming humanrobot interactions. This study not only advances our understanding of effective HRI strategies but also opens up new avenues for the practical application of robots in assisting human activities, thereby enriching the quality of human life.

Acknowledgments. This work has been carried out by grant PID2022-138453OB-I00 funded by MICIU/AEI/10.13039/501100011033 and by "ERDF A way of making Europe".

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

- Cao, H., Cooper, D.G., Keutmann, M.K., Gur, R.C., Nenkova, A., Verma, R.: Crema-d: Crowd-sourced emotional multimodal actors dataset. IEEE transactions on affective computing 5(4), 377–390 (2014)
- Costa, A., Martinez-Martin, E., Cazorla, M., Julian, V.: Pharos—physical assistant robot system. sensors 18(8), 95 – 107 (2018)
- Jackson, P., Haq, S.: Surrey audio-visual expressed emotion (savee) database. University of Surrey: Guildford, UK (2014)
- Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., Casas, D.d.l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al.: Mistral 7b. arXiv preprint arXiv:2310.06825 (2023)
- Kanda, T., Miyashita, T., Osada, T., Haikawa, Y., Ishiguro, H.: Analysis of humanoid appearances in human–robot interaction. IEEE transactions on robotics 24(3), 725–735 (2008)
- Khalaf, A.L., Abdulrahman, M.M., Al_Barazanchi, I.I., Tawfeq, J.F., JosephNg, P.S., Radhi, A.D.: Real time pedestrian and objects detection using enhanced yolo integrated with learning complexity-aware cascades. TELKOMNIKA (Telecommunication Computing Electronics and Control) 22(2), 362–371 (2024)
- Liang, C., Lu, J., Yan, W.Q.: Human action recognition from digital videos based on deep learning. In: Proceedings of the 5th International Conference on Control and Computer Vision. pp. 150–155 (2022)
- Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. Advances in neural information processing systems 36 (2024)
- Livingstone, S.R., Russo, F.A.: The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. PloS one 13(5), e0196391 (2018)

- Marin, E.G.C., Morales, C.A., Sanchez, E.S., Cazorla, M., Plaza, J.M.C.: Designing a cyber-physical robotic platform to assist speech-language pathologists. Assistive Technology 35(1), 48–55 (2023)
- Mejia-Escobar, C., Cazorla, M., Martinez-Martin, E.: A large visual, qualitative, and quantitative dataset for web intelligence applications. Computational Intelligence and Neuroscience **2023**(1), 1094823 (2023)
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of machine learning research 21(140), 1–67 (2020)
- Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533-536 (1986). https://doi.org/10.1038/323533a0, https://doi.org/10.1038/323533a0
- Tiedemann, J., Thottingal, S.: Opus-mt-building open translation services for the world. In: Proceedings of the 22nd Annual Conference of the European Association for Machine Translation. pp. 479–480 (2020)
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. arxiv 2017. arXiv preprint arXiv:1706.03762 3762 (2023)
- Vryzas, N., Kotsakis, R., Liatsou, A., Dimoulas, C.A., Kalliris, G.: Speech emotion recognition for performance interaction. Journal of the Audio Engineering Society 66(6), 457–467 (2018)
- Wu, X., Bartram, L.: Social robots for people with developmental disabilities: a user study on design features of a graphical user interface. arXiv preprint arXiv:1808.00121 (2018)

People detection on 2D laser range finder data using deep learning and machine learning^{*}

José Abrego-González, Eugenio Aguirre, and Miguel García-Silvente

Department of Computer Science and A.I. (DECSAI). Andalusian Research Institute in Data Science and Computational Intelligence (DaSCI). CITIC-UGR., University of Granada (UGR), 18071 Granada, Spain jabrego@correo.ugr.es, eaguirre@decsai.ugr.es,

m.garcia-silvente@decsai.ugr.es

Abstract. This work presents a machine learning based study on people detection using 2D Laser Range Finders (LRFs) combined with deep learning methodologies, aimed at enhancing mobile robot capabilities in various environmental conditions. The study introduces a novel integration of a monocular camera with an LRF on a mobile robot to improve the accuracy and efficiency of detecting and tracking people. By employing deep learning models such as CenterNet, the system leverages both image and 2D range data to facilitate automatic labeling of datasets, crucial for training robust classification algorithms. In order to achieve the best classifier, two experimental studies are introduced in this work. The former is carried out in a simulated environment and the latter in real-world, office-like environments. In simulations, various machine learning models are trained and evaluated, showing significant results in distinguishing human legs from other objects. The transition to realworld testing underscores the challenges and adaptations necessary to achieve high accuracy and reliability in dynamic settings. The XGBoost model emerged as the most effective classifier in our study, achieving the highest scores in accuracy, precision, recall, and F1-score, outperforming other methods across these key metrics. This work aims to advance the field of 2D LRF based people detection and also proposes a solution for real-time applications, balancing precision and computational efficiency. Experimental results from both simulated and real-world environments demonstrate the system's effectiveness.

Keywords: People detection · Deep learning · Machine learning · 2D LRF

1 Introduction

Mobile robots rely on detecting and tracking people for applications like Human-Robot Interaction (HRI), navigating crowded spaces, and safety in shared environments [20]. Various computer vision techniques using Monocular, Stereo, and

^{*} This work was made possible thanks to the support of Senacyt Panamá (Scholarship No. 270-2022-164) and Grant PID2022-138453OB-I00 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe".

RGB-Depth cameras, including deep learning methods like YOLO, have proven effective for these tasks [15]. Despite the advantages of vision systems, 2D Laser Range Finders (LRFs) are favored in social and service robots for their reliability and wide field of view, overcoming the limitations of vision sensors in adverse conditions [4].

Detecting people using 2D laser technologies can be achieved through various approaches, as outlined in the survey conducted by M. Sharif [22]. Some techniques directly process laser measurements individually as inputs for supervised machine learning algorithms. Conversely, alternative methods first cluster these measurements and then derive features to characterize these clusters. In this work, we will adopt the latter approach and propose a set of innovative features compared to those described in the specialized literature.

Emerging studies utilize deep learning for enhanced detection from sensor data, offering significant improvements in reliability [21]. Given the absence of automatic labeling tools for 2D laser data, this work explores the potential of deep learning to automate the labeling of such datasets and the use of machine learning approaches to generate efficient leg detectors, aiming to enhance efficiency and accuracy in diverse applications [14].

This work is organized as follows: Section 2 outlines the proposed system's hardware and software, emphasizing camera and LRF integration. In Section 3, deep learning methods for object detection are detailed, with a focus on the CenterNet model. Section 4 describes the mobile robot's hardware and software components. Section 5 discusses the simulated environment for leg detection using 2D LRF and the associated machine learning training. Experimental evaluations in simulated and real-world conditions are covered in Sections 5 and 6, respectively. The conclusions and some ideas on future work are commented in Section 7.

2 Description of the proposal

In the initial phase of this study, we employ a simulated environment using CoppeliaSim to rigorously test our experimental setup. This approach ensures that it performs effectively under controlled conditions designed to mirror real-world scenarios. This simulation allows us to evaluate the detection capabilities and overall system reliability without the complexity and variability inherent in physical environments. Through this simulation, we gain insights into the system's performance, providing a foundation for further development and refinement before real-world application.

The next step involves using real-world data to validate the findings from our simulations. Building on prior work in people detection, tracking [2], and automatic labeling of 2D range data [1], this proposal introduces a refined method by employing a monocular camera instead of the Kinect 1.0 sensor, avoiding the Kinect's infrared drawbacks under certain lighting conditions. The system incorporates a LRF sensor mounted 30 cm above the floor on a mobile robot, paired with a Jetson TX2 Developer Kit, which manages the camera and LRF

integration. This setup captures both image and 2D range data alongside odometry and velocity data, stored on an onboard SSD during navigation through an office-like environment.

This data supports offline deep learning analysis on powerful machines using TensorFlow 2 Object Detection API [12]. A specific focus is on the Center-Net HourGlass104 Keypoints 512x512 model [27], trained on the COCO 2017 dataset [17], to detect people in images. This model helps in keypoint detection, aiding in the identification of human legs through images, which are then correlated with 2D range data for automatic labeling. The labeled data facilitates the development of a binary classifier for detecting people's legs via machine learning algorithms.

3 Deep learning based object detection

3.1 Two and one stage approaches

Object detection methods, essential for identifying categories like people and animals in images, have advanced significantly with deep learning, particularly through Convolutional Neural Networks (CNN) [10]. These methods fall into two primary categories: two-stage and one-stage approaches.

Two-stage approaches, such as R-CNN [10] and its successors, Fast-RCNN [9] and Faster-RCNN [19], involve first extracting Regions of Interest (RoIs) and then classifying them. Innovations include Mask-RCNN [11] for simultaneous object and mask detection, and R-FCN [8], which employs position-sensitive score maps. Cascade R-CNN [5] addresses overfitting by training sequential detectors with increasing Intersection over Union (IoU) thresholds.

One-stage approaches, exemplified by YOLO [25] and SSD [18], streamline the process by directly classifying and regressing anchor boxes, eliminating the need for separate RoI extraction. Keypoint-based methods like CenterNet [28] represent objects using keypoints, which simplifies bounding box determination and avoids traditional anchor box disadvantages.

3.2 CenterNet

CenterNet utilizes a single central point in an object's bounding box for representation, regressing other properties like size and pose from image features [27]. A keypoint heatmap generated by a fully convolutional network aids in detecting object centers, and bounding boxes are predicted from these peaks. The model uses dense supervised learning for training and operates in real-time without non-maximal suppression during inference.

CenterNet not only provides excellent speed-accuracy trade-offs on the COCO dataset but also allows for multi-person pose estimation, identifying human joints as offsets from the center [6]. Various applications have demonstrated Center-Net's efficacy, such as fault diagnosis in train catenaries [7], biometric recognition [26], vehicle detection [23], and real-time person detection in surveillance [3].

For this project, a CenterNet HourGlass104 Keypoints 512x512 model pretrained on the COCO 2017 dataset [17] was selected based on its optimal balance between speed and accuracy. This model is integral to detecting people and their keypoints within our collected dataset, demonstrating its versatility and robustness in object detection.

4 System overview

The system consists of three primary hardware components. The first is a PeopleBot mobile robot equipped with an LRF SICK LMS200 [13], which has a 180° field of view and is capable of accurate measurements up to 8 meters. The second component is an NVIDIA Jetson TX2 Developer Kit, mounted on top of the LRF. This integrated unit features both GPU and CPU, optimized for high efficiency and power, supporting an onboard camera with a resolution of 640 x 480 at 30 frames per second. The Jetson TX2 operates on Ubuntu 18.04 and connects to the robot's sensory system via USB. The third component is a laptop, positioned on the robot and connected to the Jetson TX2 via Ethernet, serving as the user interface.

The software architecture of the system is developed in C++, utilizing Aria and ArNetworking libraries provided by the robot manufacturer for programming and network communication, although only Aria is needed due to the direct wired connection of the Jetson TX2 which replaces the robot's original onboard computer. The Jetson TX2 is powered by a LiPo battery, similar to those used in drones, while the robot operates on a standard plumb battery system. OpenCV library manages the image processing, and both laser measurements and images are saved directly to an SSD connected to the Jetson TX2.

5 Leg detection using 2D LRF in simulated environments

To evaluate the effectiveness of various techniques for detecting people using 2D LRF data, a simulated environment was utilized. CoppeliaSim was chosen for its straightforward integration with several programming languages, including Python, which facilitated the simulation process.

Python was the primary programming language employed in this study due to its widespread adoption in research and data science. Its clear syntax and a comprehensive array of specialized libraries support efficient, complex data analysis and model development. Key libraries used include NumPy for numerical computations, Pandas for data manipulation, scikit-learn for machine learning, and OpenCV for computer vision. This combination of accessibility and robust functionality makes Python especially suitable for conducting data-driven research efficiently.

5.1 Data Collection Process

The data collection process for leg detection using a 2D LRF was meticulously designed within the CoppeliaSim simulation environment. Three unique scenar-
ios were crafted to closely replicate real-world dynamics and interactions. These scenarios include a single person moving along a predefined path, a static group of people, and an assortment of non-human objects that mimic the shape of human legs.



Fig. 1: Multiple scenarios for data collection: (a) Single person following a path; (b) Multiple people in stationary positions; (c) Objects-only scenario.

The detailed process for collecting the 2D LRF data is as follows:

- Single Person Scenario: The scene incorporates only the Pioneer robot model and a person model that follows a predefined path (Bill on track). The robot remains stationary while the person navigates a path composed of consecutive S shapes, initially in a perpendicular direction, which then shifts to a parallel orientation. The exact path is illustrated in figure 1a. This scene is designed to capture samples of legs in motion. This specific scenario is crucial as leg detection of a moving person appears slightly distorted due to the sampling speed of the 2D LRF.

- Multiple Person Scenario: In this scene, 25 person models are used, comprising 13 standing models (Standing Bill) and 12 sitting models (Sitting Bill). Each model is randomly rotated to ensure a variety of perspectives before data collection begins. The robot moves along a predefined path, marked by yellow arrows in figure 1b, while the person models remain stationary.
- Objects Only Scenario: This setup is filled with diverse objects including walls, rectangles, and cylinders of various sizes, arranged to simulate a complex environment. The goal is to gather non-leg samples, enhancing the dataset's diversity. The robot navigates a predefined path, indicated by yellow arrows in figure 1c, during data collection.

After a comprehensive data collection process, we now turn our attention to the next section where we will delve into the specifics of how we have applied clustering to our collected data.

5.2 Clustering

In our simulated environment, we chose to use settings that replicate those of the physical LRF SICK LMS200 laser sensor. This decision ensures that the simulated robot's operational characteristics align closely with those of its realworld counterpart. Specifically, the simulated Pioneer robot employs a 2D LRF that provides a 180-degree field of view with an angular resolution of 0.5 degrees, positioned approximately 30 cm above the ground.



Fig. 2: Outline of a person's leg in both simulated (a) and real world (b) environments.

When a standing person is scanned by this LRF, the image appears as two closely spaced semi-circumferences, as illustrated in Figure 2. Each sensor scan generates 361 polar coordinates, which are then transformed into Cartesian coordinates for further analysis. One of our primary objectives is to discern features within the sensor's field of view, particularly focusing on the lower portion of a person, such as legs. To approach this challenge, we adopted a binary classification framework. The goal is to categorize the data into two groups: representations of human legs and all other objects. While this method enhances our ability to distinguish between various shapes, it is important to recognize the limitations inherent in using laser-based technology. For instance, objects like table legs or other cylindrical items that mimic the dimensions of human legs can pose challenges for accurate differentiation. Thus, while our approach provides a structured way to analyze the sensor data, the precision of distinguishing between very similar shapes solely through this method can vary.

Given the variable nature of the cluster sizes in our dataset, we incorporated a variety of features for cluster characterization. These features are drawn from established practices in related research and general techniques employed for polygon characterization.

5.3 Features extraction

To prepare data samples for machine learning tasks, we extract a comprehensive set of geometric features from the clusters, expanding upon successful methods like the Leigh's detector [16]. The comprehensive set of features utilized to describe the clusters includes:

- Depth: Measurement of the cluster's extent from the front to the back.
- Width: Measurement across the widest part of the cluster perpendicular to the depth.
- Perimeter: The total length around the boundary of the detected cluster.
- Radius: The radius of the circle that best fits the point cloud as determined by the Taubin fit.
- Sigma: The mean squared error (MSE) between the fitted circle and the actual points in the cloud, indicating the fit's accuracy.
- Area: Area of the polygon calculated using the shoelace formula.
- **Distance:** Distance from the centroid of the point cloud to the 2D LRF, providing a spatial reference.
- Number of points: Number of points comprising the cluster, reflecting its density and complexity.
- Angles: Sum of the internal angles of the polygon, which helps in understanding the geometric structure.

Given the typical appearance of a leg in laser-based imagery resembling semicircles, the algebraic circle fit method developed by G. Taubin [24] is applied. This technique allows us to identify the circle that most accurately represents the shape of the point cloud.

With a comprehensive set of features defined for each cluster, we them proceed to the model training phase.

5.4 Model Training

The next phase of our research involves training various machine learning models, primarily selected from the scikit-learn library. Our selection encompasses a range of robust algorithms including Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest, and AdaBoost. Additionally, XG-Boost, though not part of scikit-learn, has been included due to its efficacy in handling large and complex datasets.

The simulated environment dataset comprised 18,890 positive samples (60%) and 12,536 negative samples (40%), revealing an initial class imbalance. To address this, we balanced the dataset by randomly sampling from the positive class to match the size of the negative class, using a fixed seed for consistent results. This adjustment ensures equal representation of both classes, facilitating more effective and unbiased model training.

We allocated 75% of this balanced dataset for training and the remaining 25% for testing. The splitting process was conducted through a stratified random selection using a predefined seed to ensure the reproducibility of our results. To optimize the models, we utilized a grid search coupled with 5-fold cross-validation to determine the best hyperparameters for each algorithm. This structured approach aims to maximize the predictive accuracy and reliability of our classification models.

With the model training complete, we now move on to evaluate their performance. In the next section, we will present key metrics such as accuracy, precision, recall, and the f1-score, providing a clear overview of the different models' effectiveness.

5.5 Performance metrics

iou in solu.				
Model	Accuracy	Precision	Recall	F1-Score
XGBoost	0.994	0.994	0.993	0.994
Random Forest	0.992	0.990	0.995	0.992
KNN	0.992	0.991	0.993	0.992
SVM (kernel=rbf)	0.990	0.990	0.990	0.990
AdaBoost	0.981	0.977	0.985	0.981
SVM (kernel=linear)	0.966	0.946	0.989	0.967
SVM (kernel=poly)	0.963	0.957	0.969	0.963

Table 1: Performance metrics of various classification models evaluated on the test split of the dataset, sorted by fl-score. The highest scores for each metric are highlighted in bold.

The analysis of various machine learning algorithms on the dataset highlights distinct performances, suitable for different practical applications. The XGBoost exhibited superior metrics (accuracy, f1-score, precision each at 0.994, and recall

at 0.993), suggesting its robustness in achieving high predictive accuracy. It is followed closely by the KNN and Random Forest, both showcasing high accuracy and f1-score around 0.992, with Random Forest slightly outperforming in recall (0.995). For detailed performance metrics of each model, see Table 1.

The SVM with an RBF kernel also performed commendably, maintaining balanced metrics around 0.990, although other kernel types showed reduced effectiveness. The AdaBoost and SVM with a linear kernel demonstrated moderate success.

Moving forward from our simulation results, the next phase of our study involves applying the best models to a real-world setting to validate their practical efficacy. This step is crucial for transitioning from a controlled experimental environment to actual operational scenarios, where variables and conditions are more dynamic and unpredictable.

6 Real World Environment

In the real-world phase, we leverage an automatic labeling process facilitated by deep learning models specializing in pose estimation. This approach aims to enhance the accuracy of data annotation, which is pivotal for training and validating our machine learning models under real-world conditions.

6.1 Data Collection Process

Data was gathered using the PeopleBot mobile robot equipped with an LRF SICK LMS200. The robot captured multiple sequences, collecting both 2D laser data and visual images via an onboard camera on the Jetson TX2 development board. The scenarios involved both static and dynamic elements: the robot remained stationary while multiple individuals walked in its vicinity, and in other tests, the robot moved, simulating the task of following a designated person. This movement was managed using an industrial joystick. Although the focus of this study does not include robot control—which will be explored in future work—the recorded scenarios were designed to reflect the expected operational conditions of the mobile robot in real-world settings.

The primary environments for these recordings were office-like spaces, including corridors, office rooms, and larger indoor areas such as conference rooms and hallways. This setup aimed to simulate typical interactions and navigational challenges the robot would face in a working environment.

In total, 9,241 frames across three different sequences were captured with the robot in stationary positions, and 6,320 frames in five sequences where the robot was manually controlled to mimic the behavior of following a person.

Having described the data collection process and the operational settings, we now turn our attention to the methodology for processing this data. The next section delves into the automatic data labeling process, which utilizes pose estimation techniques.

6.2 Automatic labeling using bounding box and keypoints

Leveraging 2D range data for leg detection involves machine learning techniques applied to datasets that ideally represent realistic operational scenarios, as discussed in Section 1. Our approach uses a pre-trained CenterNet HourGlass104 Keypoints (CHK) model to identify bounding boxes and keypoints for legs, facilitating the automatic labeling of 2D range data. The process integrates the detection of people in images from an onboard camera with the localization of corresponding laser points.



Fig. 3: (a) Keypoints diagram of CenterNet HourGlass104 Keypoints. (b) Bounding box and keypoints supplied by CenterNet Hour Glass104 Keypoints detector.

Bounding boxes and keypoints for human legs are defined by specific keypoints: kp_{13}^i and kp_{15}^i for the left leg, and kp_{14}^i and kp_{16}^i for the right leg, as illustrated in Fig. 3. 2D laser points are clustered using the jump distance algorithm, with valid clusters transformed and projected onto image coordinates using the robot's camera calibration parameters.

The calibration process involves capturing images of a checkerboard pattern from various angles to compute the camera's intrinsic (focal length, optical centers) and extrinsic (position and orientation in relation to the robot) parameters. This alignment is essential for enabling a seamless overlay of laser data onto the visual images and facilitates the accurate identification and labeling of leg clusters based on the proximity of laser points to the keypoints on the images.

The process begins by projecting the 2D LRF data onto the image captured by the RGB camera using a transformation matrix. This alignment allows us to overlay the LRF data onto the corresponding visual content accurately. Once aligned, the CenterNet model identifies key points on the person, specifically kp_{13}^i , kp_{15}^i , kp_{14}^i , and kp_{16}^i , which correspond to the left knee, left foot, right knee, and right foot, respectively.



Fig. 4: (a) Projection of 2D laser range data to colour image. (b) Assignment of the projections of clusters to detect legs.

Considering the LRF primarily captures data at shin height and there are no direct equivalent keypoints, a new reference point for each leg is created by calculating the midpoint between the knee and foot keypoints. The distance between this newly established midpoint and the nearest LRF point cluster is then measured. If this distance is within a predefined threshold and the rest of the points in that cluster are nearby, then the cluster is marked as a positive leg detection, indicating the presence of a person's leg.

Clusters not identified as legs are labeled as negative samples, representing background objects. This selective filtering aims to ensure the data used for training and validation is both relevant and accurately labeled.

The process is visualized in Fig. 4a which illustrate the integration of image and laser data, and Fig. 4b showing the assignment of clusters to detected leg keypoints.

With our data now accurately labeled using the automated process, we proceed to characterize the data using the techniques outlined in section 5.2 before moving to the next phase where once again we will train a selection of machine learning models. This stage will leverage adjustments and enhancements, informed by the insights gathered during the prior phases.

An experimental study, outlined in the following section, selects the optimal machine learning algorithm for classifying these features, ensuring robust leg detection using 2D range data.

6.3 Model training

Following the evaluation criteria outlined in section 5.5, we narrowed down the selection to models that achieved an F1-score of 99% or higher. This criterion ensured that only the most accurate models were considered for training on real-world data. This select group includes: XGBoost, Random Forest, KNN and SVM (with RBF kernel).

Given the close performance results of these models, a new metric, inference time, has been introduced to find the best trade-off between computing time and performance. This metric measures the speed at which a model can process input data and return a result, an essential factor for real-time applications. The inclusion of inference time helps assess the practicality of deploying these models in scenarios where response speed is critical.

After characterization, we obtained a dataset consisting of 19,632 samples, balanced evenly across both classes. The same methodology outlined in section 5.4 was used to split the dataset, and also for selecting the best hyperparameters for the different models.

With our dataset prepared and models optimized, we then proceed to a detailed evaluation. In the upcoming section, we will analyze model effectiveness using both traditional accuracy metrics and the newly introduced inference time, ensuring a better understanding of their potential for real-world applications.

6.4 Performance metrics

To comprehensively assess the performance of our models, we focused on calculating the mean inference time. This metric represents the average processing time required for 1,000 samples across 19 distinct splits of our complete balanced dataset. Additionally, we calculated the standard deviation to evaluate the variability in processing times across these splits.

These tests were conducted using a high-performance laptop equipped with an Intel Core i7-12650H processor, featuring a 24 MB L3 cache, speeds up to 4.70 GHz, 10 cores, and 16 threads. The system also includes 16 GB of DDR5 RAM operating at 4800 MHz across two modules. The laptop also features an NVIDIA GeForce RTX 4070 mobile GPU; however, this component was not utilized in the current stage of testing as our analysis was confined to machine learning models that did not leverage GPU acceleration.

Table 2: Performance metrics of various classification models evaluated on the test split of a real world dataset, sorted by f1-score. The highest scores for each metric are highlighted in bold.

metrie die inginighted in bold.										
Model	F1-Score	Accuracy	Precision	Recall	Mean (ms)	STD (ms)				
XGB	0.9944	0.9944	0.9946	0.9943	2.0	0.4				
$\operatorname{RandomForest}$	0.9941	0.9941	0.9953	0.9928	23.1	1.6				
KNeighbors	0.9930	0.9930	0.9946	0.9914	57.4	4.8				
SVC (kernel=rbf)	0.9913	0.9914	0.9960	0.9867	343.1	12.7				

In the real-world dataset, the evaluation of machine learning algorithms (Table 2) has yielded results that closely align with those obtained in the simulated environment, demonstrating a strong consistency across different experimental setups. The XGBoost model has continued to exhibit exceptional performance, achieving the highest f1-score of 0.9944, which correlates precisely with its accuracy. This high degree of accuracy and precision (0.9946) coupled with a nearly identical recall (0.9943) underscores its robustness in handling real-world data. Additionally, the algorithm benefits from a notably efficient execution time, averaging just 2.0 milliseconds per instance with a minimal standard deviation of 0.4 milliseconds, highlighting its suitability for applications requiring high-speed data processing.

Random Forest also shows excellent performance, slightly surpassing XG-Boost in precision but with a longer execution time. However, its capacity for easy explainability may render it particularly valuable in scenarios where understanding model decisions is crucial, despite the slower performance.

K-Nearest Neighbors (KNN) and the Support Vector Machine (SVM) with an RBF kernel, while robust, are slower and slightly less effective in terms of recall and overall f1-score compared to XGBoost and Random Forest. The significant processing time of the SVM, in particular, could be a drawback in rapid decision-making environments. These findings confirm the suitability of XGBoost and Random Forest for practical applications, emphasizing the need to balance performance metrics and operational requirements when selecting algorithms.

For an in-depth understanding of the models configuration settings, we present the specific parameters used for the three top-performing algorithms in our study:

- Random Forest is set with an entropy criterion, a maximum depth of 12, and 100 estimators. The maximum features are determined by the square root of the total number of features, which optimizes the diversity and computational efficiency of the model.
- XGBoost a maximum depth of 6 with 200 estimators to balance learning capacity and prevent overfitting, ensuring high precision and recall in its performance metrics.
- **KNN** employs the ball tree algorithm with 11 neighbors and weights by distance, using the Manhattan distance metric (p=1). This configuration enhances sensitivity to local data structures, beneficial for our complex dataset.

7 Conclusions

The integration of deep learning-based pose estimation with traditional 2D Laser Range Finders (LRFs) has significantly enhanced mobile robots' ability to detect human presence. The automated fusion of image and 2D range data streamlines the creation of accurate training datasets, crucially reducing the manual effort and time involved in labeling.

Experimental results validate the effectiveness of the proposed system, particularly highlighting the performance of the XGBoost model, which demonstrates high accuracy and reliability in both simulated and real-world settings. This model proficiently distinguishes between human figures and other objects, ensuring practical utility in operational environments. Future work will focus on developing people tracking capabilities using the predictions generated by models trained in this study. This advancement aims to enhance the real-time tracking and interaction capabilities of mobile robots in complex environments, further bridging the gap between laboratory settings and real-world applications.

References

- Aguirre, E., García-Silvente, M.: Using a deep learning model on images to obtain a 2d laser people detector for a mobile robot. International Journal of Computational Intelligence Systems 12(2), 476-484 (Jan 2019). https://doi.org/10.2991/ijcis.d. 190318.001
- Aguirre Molina, E., García Silvente, M., Pascual, D.: A multisensor based approach using supervised learning and particle filtering for people detection and tracking (2016). https://doi.org/10.1007/978-3-319-27149-1 50
- Ahmed, I., Ahmad, M., Rodrigues, J., Jeon, G.: Edge computing-based person detection system for top view surveillance: Using CenterNet with transfer learning. Applied Soft Computing 107, 107489 (05 2021). https://doi.org/10.1016/j.asoc. 2021.107489
- Benedek, C.: 3D people surveillance on range data sequences of a rotating lidar. Pattern Recognition Letters 50, 149–158 (2014). https://doi.org/10.1016/j.patrec. 2014.04.010
- Cai, Z., Vasconcelos, N.: Cascade R-CNN: Delving into high quality object detection. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6154-6162 (2018). https://doi.org/10.1109/CVPR.2018.00644
- Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1302-1310 (2017). https://doi.org/10.1109/CVPR.2017.143
- Chen, Y., Song, B., Zeng, Y., Du, X., Guizani, M.: A deep learning-based approach for fault diagnosis of current-carrying ring in catenary system. Neural Computing and Applications pp. 1-13 (07 2021). https://doi.org/10.1007/s00521-021-06280-4
- Dai, J., Li, Y., He, K., Sun, J.: R-FCN: Object detection via region-based fully convolutional networks. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 29 (2016)
- Girshick, R.: Fast R-CNN. In: 2015 IEEE International Conference on Computer Vision (ICCV). pp. 1440-1448 (2015). https://doi.org/10.1109/ICCV.2015.169
- Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition. pp. 580-587 (2014). https: //doi.org/10.1109/CVPR.2014.81
- He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 2980-2988 (2017). https://doi.org/10.1109/ICCV.2017.322
- Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., Murphy, K.: Speed/accuracy trade-offs for modern convolutional object detectors. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3296-3297. IEEE Computer Society, Los Alamitos, CA, USA (2017). https://doi.org/10.1109/CVPR.2017.351

- 13. Intelligence, S.S.: Sick sensor intelligence, LMS200 (2002), http://www.mysick.com
- Kim, J.J., On, B.W., Lee, I.: High-quality train data generation for deep learningbased web page classification models. IEEE Access 9, 85240-85254 (2021). https: //doi.org/10.1109/ACCESS.2021.3086586
- Lafuente-Arroyo, S., Martin-Martin, P., Iglesias-Iglesias, C., Maldonado-Bascon, S., Acevedo-Rodriguez, F.J.: RGB camera-based fallen person detection system embedded on a mobile platform. Expert Systems with Applications p. 116715 (2022). https://doi.org/10.1016/j.eswa.2022.116715
- Leigh, A., Pineau, J., Olmedo, N., Zhang, H.: Person tracking and following with 2d laser scanners. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). pp. 726-733 (2015). https://doi.org/10.1109/ICRA.2015.7139259
- Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014. pp. 740–755. Springer International Publishing, Cham (2014)
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single shot multibox detector. In: European conference on computer vision. pp. 21-37. Springer (2016)
- Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 28 (2015)
- Rubagotti, M., Tusseyeva, I., Baltabayeva, S., Summers, D., Sandygulova, A.: Perceived safety in physical human-robot interaction-a survey. Robotics and Autonomous Systems 151 (2022). https://doi.org/10.1016/j.robot.2022.104047
- S. Mohamed, I., Capitanelli, A., Mastrogiovanni, F., Rovetta, S., Zaccaria, R.: Detection, localisation and tracking of pallets using machine learning techniques and 2d range data. Neural Computing and Applications **32**, 8811–8828 (07 2020). https://doi.org/10.1007/s00521-019-04352-0
- Sharif, M.H.: Laser-based algorithms meeting privacy in surveillance: A survey. IEEE Access 9, 92394-92419 (2021). https://doi.org/10.1109/ACCESS.2021. 3092687
- Sun, Y., Li, Z., Wang, L., Zuo, J., Xu, L., Li, M.: Automatic detection of vehicle targets based on centernet model. In: 2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE). pp. 375-378 (2021). https://doi.org/10.1109/ICCECE51280.2021.9342498
- Taubin, G.: Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation. IEEE Trans. PAMI 13, 1115-1138 (1991)
- Vijayakumar, A., Vairavasundaram, S.: Yolo-based object detection models: A review and its applications. Multimedia Tools and Applications (Mar 2024). https://doi.org/10.1007/s11042-024-18872-y
- Yuan, L., Mao, J., Zheng, H.: Ear detection based on CenterNet. In: 2020 IEEE 2nd International Conference on Civil Aviation Safety and Information Technology (ICCASIT. pp. 349-353 (2020). https://doi.org/10.1109/ICCASIT50869.2020. 9368856
- Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. In: arXiv preprint arXiv:1904.07850 (2019)
- Zhou, X., Wang, D., Krähenbühl, P.: Objects as points (2019). https://doi.org/10. 48550/arXiv.1904.07850

Lightweight Automated Vehicle Simulator for Studying Drivers Reactions in Real Scenarios

Mohammed Bendahmane¹ and Lluís Ribas-Xirgo^{1[0000-0003-1419-0485]}

Dept. Microelectronic and Electronic Systems Universitat Autònoma de Barcelona School of Engineering, Campus UAB, 08193 Bellaterra, Spain Lluis.Ribas@uab.cat

Abstract. Traffic reports, naturalistic observations and in-lab tests are carried out to determine the influence of driver's distractions in car crashes and to develop technology to detect and prevent them from happening. Paradoxically, driver's distractions are more prone to occur with automated driving vehicles, which might turn into cancelling some of their advantages. One of the best strategies to cope with this problem is education. In this work we propose a simulation environment to design laboratory experiments to study driver's distractions as well as simple games to train drivers to take over the control of the vehicle when required. The resulting simulator enables drivers to experience driving partially automated cars in their neighborhoods or in places of their choice with different levels of events to check their ability in taking over the control when required.

Keywords: Autonomous vehicles · Car-driving game · Drivers' distraction · Simulated driving.

1 Introduction

Driving on highways or in the neighborhood in a sunny day with few other vehicles on the road are situations where our sense of continuity makes us focus our attention on non-driving related tasks or get into drowsiness. Even if we drive in complex situations with denser traffic or pedestrians around, which may make unpredictable movements, we trust on paying attention to these because of our natural trend to focus on changes in the environment. Additionally, we have an extra confidence on correctly dealing with these situations if our vehicle has some kind of advanced driving system (ADS). Unfortunately, as with our feeling that vehicles and pedestrians have some inertia in their movements, we extrapolate that the ADSs rightly react to any potentially hazardous situation, but we might be in trouble when they ask us to take over the control of the vehicle in the shortest possible time.

In this work, we have designed a simulation platform so that you can train in these situations on the streets of your neighborhood, or the place you prefer, and so that the data collected with the experiments can be used to improve the driving environment, both that of vehicle and the roads on which it travels. Although there are some games that let players drive on Google maps and the like (e.g. [https://framesynthesis.com/drivingsimulator/maps]), the interaction with the scenario is too elementary and cannot be used to measure data about the driver-players' behaviors. In the opposite side, there are realistic virtual reality games based on actual video footages [https://driving-tests.org/driving-simulator] or simulated environments [https://driving.com] which deliver a good immersive experience to the users at the cost of computing power and with limited number of scenarios. The same is true for simulators oriented to testing self-driving cars [1], as they require accurate representation of the 3D

world so that the simulated sensors can get data of the same quality than their real counterparts. This is the first work where simulated traffic environments come from real

This is the first work where simulated trainc environments come from real maps and the simulation is computationally inexpensive, like many of the psychometric driving test tools, with the addition of including the modeling of partially autonomous driving of the vehicle. Differently from many current driving simulators that try to convey realistic scenes to the users, the proposed solution relies on Gestalt principles to fill in the blanks in the simplified visual environment and on the user-engagement effect of the situated game. In this sense, it provides a generic, non-compute-intensive tool for driving schools [2] and researchers. Like the psychotechnical test drive games and tools to assess and train coordination, reflexes and prediction abilities [3, 4], our solution can help doing the same for partially autonomous cars.

1.1 Literature Review

Most of us suffer significant impairment when using cell phones while driving [5] and are less aware of road hazards when driving partially automated vehicles (PAVs) [6]. In effect, drivers, particularly the novice [7], trust on advanced driverassistance systems (ADAS) and ADS [8] and get distracted easily. These distractions and inattention might cause failure to appropriately and timely respond to potentially dangerous events while driving, thus being necessary to study different interventions to mitigate these adverse effects of non-driving related tasks (NDRT) and drowsiness. Unfortunately, their impact on driving performance and their contribution to car crashes are difficult to determine because of different definitions used to classify accidents and because not all safety-critical events end up in a car crash. Anyway, distraction and inattention might be the cause of 10% to 25% of accidents, which could have been avoided with the use of appropriate warning systems and by specific drivers' training [9, 10].

Experimentation allows not only to understand the mechanisms that lead drivers to these distraction episodes and inattention but also to test systems for their detection and early warning, as well as to develop instructional materials and exercises for drivers. For example, it is possible to design a better HMI to mitigate the adverse effects of distraction by investigating visual attention while driving [11]. Some studies are carried out with actual vehicles but, for safety reasons, most of them are done on simulators. Anyway, the main problem is they are limited to 30-40 participants driving actual cars in controlled areas or simulated vehicles in simple environments like rural areas [12], which might not be representative of the general population in all sort of situations.

Although simulators have been improved in detail and capability to represent complex scenarios of suburban and urban areas, their hardware requirements hinder their use in studies with larger groups of participants.

Regardless of the experimentation environment, measuring driver's distraction is complex because it depends on observations with cameras [13] and other sensors, including those that measure driving performance (steering, throttle, speed, ...) [14], and on software methods to determine the onset of the distraction events like support vector machines [14, 15] and neural networks [16].

In this regard, the Insurance Institute for Highway Safety developed a rating system [17] that evaluates the safety of ADSs and their driver monitoring systems (DMS) based on how good they are at identifying driver distraction, how they alert drivers once the distraction has been identified and also whether they employ any safety measures when the driver does not react to the alerts.

There have been attempts to use driver state assessment technology to keep the driver 'in-the-loop', i.e. to keep drivers' attention to driving [18], and even to safely stop the vehicle if the drivers fail to react to systems' alerts [19], but holistic solutions must consider education, too [20].

Drivers' ability to understand partially automated driving and to know its boundaries might make them timely taking over control of the vehicles, which might avoid crashes. Note that drivers are responsible for the vehicle even in situations where automation required unreasonably rapid reactions from them [21]. Thus, apart from education, it is required that automation is validated against a broad spectrum of possible users.

A recent literature survey [22] concludes that it is easier that drivers get engaged in NDRT while driving vehicles with ADAS and ADS, that the older drivers tend to trust less on these systems than the younger ones, and that the main solution is that drivers know better how these systems work and how to interact with them.

Thus, the main problem is the training/education of drivers, which requires appropriate materials and frameworks, including simulators. Following this idea, a driving simulator of an autonomous vehicle is a key element in any drivers' training/experimentation framework. These driving simulators can be built on top of a game engine that integrates simulation of physical environments with graphics (Section 2). Achieving a great level of realism would also make the result driving games highly demanding in terms of computational power and hardware. In this work we propose a less resource-demanding simulation software with the ability to generate scenarios from actual roadmaps to compensate for the lack of realism as for players' engagement in the game.

Additionally, this "situated experimentation" allows to study the effects of the traffic infrastructure in drivers' distraction. Note that, for instance, curves make drivers be more attentive [7], that there are other factors that contribute to drowsiness than supervising ADAS like the road conditions and environment, and that drowsiness might be worse than distractions [23].

The 3D scenarios are generated from areas of interest taken from Open-StreetMap (Section 3) and the Godot game engine is used to play the simulation of an autonomous vehicle that hands the driving control to the user in different conditions while monitoring its reactions (Section 4). The result framework is a proof of concept for this software, which can be targeted to a larger number of people than with more demanding, though realistic, simulators.

2 Game Design and Architecture

In this work, we have created a simple driving-simulation game for users to experience driving vehicles with up to the level 3 of automation [24] and for providing researchers with a tool to study the effects distractions and investigate about systems to minimize their adverse consequences on driving.

The player drives an autonomous vehicle in a 3D world with vehicles and pedestrians roaming the streets. The vehicle goes to a random destination point in the simulated world and, at random points in time, hands the control to the user, which then must react in the shortest time possible to prevent an accident. Parameterized test scenarios allow studies of drivers' behaviors in a broad range of 3D worlds and traffic conditions. Data related to the users' interactions, specifically their awareness and reaction speed, are collected during simulation an can be further analyzed by experts to gain a better understanding of the problem, improving onboard HMI, modifying the road infrastructure, *et cetera*.

The two main components of the simulation platform are the 3D world and the game of driving a car in it, with other cars and pedestrians (Fig. 1). The data to build the simulated world is taken from OpenStreetMap (OSM) [https://www.openstreetmap.org] and the game is run on the Godot engine [https://godotengine.org/] so the result platform is free to use and does not require much computational power. However, to make the simulation more appealing and immersive to the users, it is interesting to add as many details, such as non-essential objects, textures, or visual effects to the world, as possible.

The Godot game engine is an open-source game engine that provides with all the necessary tools to create a 3D environment and scripting capabilities for all the functionality that our simulation needs. Godot's main scripting language GDScript is an easy-to-use but powerful scripting language in which to describe all kinds of 3D manipulation and implement all the functionality that the simulation might need [25].

However, in other applications, particularly for developing, training, and validating ADS, detailed 3D worlds are required. In this case, you could use, for instance, the open-source CARLA simulator [26], which is built upon Unity [https://unity.com/]. This simulator can also be adapted to create a framework like the one we propose in this work by making the 3D world coarser, but we have opted for Godot because of its simplicity to install and use.



Fig. 1. The game uses the Godot engine to run a GDScript program that simulates a 3D world, extracted from an area of interest taken from OpenStreetMaps by OverPassAPI and built with OSM Buildings, wandering pedestrians and autonomous vehicles, including the primary or *ego* vehicle.

OSM is a free wiki world map from which our simulation game takes the data to create a 3D world of the area of interest where players will drive the *ego* vehicle. The XML-type file extracted from OSM (".osm") [27] is further processed to build a suitable representation of the 3D world.

Differently from the main OSM API, the Overpass API [28] allows optimized reads of the OSM database that enable querying great amounts of data in a shorter time, which makes the extraction of the area of interest within our simulation framework easier. The corresponding 3D worlds are generated by OSM Buildings [29], which creates a layer of buildings rendered in 3D using data obtained from OpenStreetMap. These buildings contain all the basic metadata relating to their properties, which might be used to change shapes and aspects.

Although this aspect is out of the scope of this work, note that the simulator must record data about games and users' performance for further analysis. Anyway, the main contribution of this work is the proof of concept that a computationally inexpensive simulation game can be used both in research about drivers' distractions and inattention and in elaborating drivers' training material.

3 Generation of 3D Worlds

As mentioned before, the OSM 2D map data is parsed to extract all the relevant information, which then would be manipulated and used to generate all the different 3D shapes of the environment. This geometry consists of 3D meshes of roads and buildings which can be placed in the 3D world according to the coordinates from the OSM map. Additionally, a graph of the area of interest is created from the waypoints in its roads so that a specific GPS module can compute paths from current vehicle position to any given destination. These paths can be used both to direct drivers to chosen spots and to tell ADS how to move the vehicle node to node until the target position is reached, thus acting as a fully autonomous vehicle. In this last mode, the vehicle can ask the user to solve complex situations or to take the control in the vicinity of pedestrians or cars.

3.1 Parsing OSM Data

OSM data can be downloaded directly from its website by choosing the region of interest, provided that the resulting file does not exceed 50000 nodes, which is enough for the purposes of this work. Anyway, this limitation is overcome by using, e.g., the Overpass API, which allows for up to 300 MB of uncompressed downloads. These *nodes*, which should not be confused with road or graph nodes, identify geolocation data of objects, and are related to other nodes by *ways*, which, at their turn, define shapes of compound objects such as buildings or roads.

In this version of the game, only objects that are tagged as "highway" or "building" are considered. To build the 3D scene, node latitudes and longitudes are converted into scene (x, y) coordinates and other object attributes are used to create the scene objects accordingly. For instance, to distinguish between roads and sidewalks, to set the number of lanes in a road, or to construct buildings with its actual number of levels.

3.2 Generating 3D Meshes

The 3D scene is made of a collection of 3D meshes that are imported from arrays of vertices that define the shapes of the corresponding objects. These vertex arrays are arranged so that every three consecutive vertices make up a triangle. The final meshes that represent paths and buildings are created with the appropriate ArrayMesh class methods from the Godot engine's library. Each final mesh, among other attributes, has an $ARRAY_VERTEX$ with the (x, y, z) coordinates of the vertices of the object that is added to the 3D scene of the game.

Buildings are derived from OSM file objects tagged as "building". The area that they take is described by a set of nodes that draw a polygon which is projected in heigh so that a wall is added every two consecutive nodes, and a roof is created on top with the same shape than the polygon at the bottom floor.

The walls and roofs of buildings are decomposed into triangles so create the mesh for the whole. While the walls are rectangular and easily split into two triangles, the roofs are partitioned into triangles by using a specific method of the Godot's *Geometry* class. Thus, each building is a set of triangles stored in a vertex array which translates to an instance of a mesh through *ArrayMesh* class' object creation method, and the corresponding object is added to the 3D scene.

Paths are represented as sequences of rectangles between any two consecutive nodes and each rectangle is divided into two triangles so to generate a mesh object in the same way as for the buildings.

Pathway meshes (Fig. 2.a) are combined with building meshes to get a basic 3D world (Fig. 2.b) which can be added details afterward.

Note that paths are created independently of buildings. In the current version, buildings that are crossed by paths act as obstacles, but they will be modified to create tunnels so that vehicles can go through them.



Fig. 2. The objects that represent paths (a) are combined with the buildings to create a basic 3D world (b).

3.3 Road graph

The ego vehicle and the rest of vehicles in the game must be able to move autonomously. In this case, actual ADS cannot be used because the 3D world is not detailed enough for sensors to produce realistic data, and because it will take too much computational power. Therefore, it is replaced by a simpler pathfollowing mechanism that will use a sequence of nodes of the *road graph*.

This road graph is made of all nodes of the *drivable* path elements in the OSM file of the region of interest. A path is drivable when is of type "highway" and the value associated with this type allows vehicles drive through. Nodes of the graph are connected when they are consecutive in a path. The resulting edges are tagged with the distance between their end nodes, so that the simulated GPS can use them to generate the shortest paths from the vehicle's current node to the destination.

4 Game Setup

The user can select which area it is interested in and run the game to drive a vehicle manually or in a partial or fully autonomous mode. In the two last cases, the driver must choose a destination. In the first case, the driver may also choose a destination and get directions from the user interface. In fact, the HMI simulates a GPS module so the users can pick up destinations within the area of interest and, as for the manual mode, it lets them to drive the vehicle.

Therefore, the key element of the game is an autonomous *ego vehicle* provided with and HMI that includes a simulated GPS module. The other vehicles in the 3D world are fully autonomous and move around going to random destinations. Additionally, there are pedestrians that might cross streets, eventually. These elements altogether enable mimicking real-life situations to test and evaluate the driver's attentiveness, namely cases where he or she avoids accidents and/or takes over the control of the ego vehicle.

Although the game stores user reaction times for further analyzes, it also may offer immediate feedback to the user by showing reaction times and scoring its overall behavior.

In the following we shall describe all these components of the game.

4.1 Vehicles

All vehicles share the same 3D model [30], which not only includes an external view (Fig. 3, top right) but also a dashboard (Fig. 3, top left). The body of the vehicle is defined as an specific class, *VehicleBody*, within Godot. This class is a sub-class of the *RigidBody* class, whose objects are controlled by the physics engine to simulate the behavior of physical objects, thus they can collide to each other and react to physical forces.

In addition to the vehicle's main body, the VehicleBody class defines the wheels of the vehicle. The front wheels are used for steering and the rear wheels for accelerating and braking. Each pair of wheels comes with a set of variables that can be accessed and manipulated in order achieve the desired movement: Variable steering have values that range from -1 to +1 that determine the orientation of the front wheels, and variables f_e (engine force) and f_b (brake) are positive values that determine the acceleration and the braking of the vehicle.

The engine force determines how much torque is applied to the rear axle in each cycle of Godot's physics engine in terms of the throttle (a) position and current rotational speed w in RPM:

$$f_e = a \cdot \tau_{max} \cdot (1 - \frac{w}{w_{max}}) \tag{1}$$

where τ_{max} is the maximum torque that can be applied by the engine to the wheels' axle and w_{max} , the maximum speed in RPM. Thus, increasing the value of the *a* results in the rear wheels turning faster and the vehicle speeding up towards the direction it is facing to.

Variable f_b determines how much force is applied to oppose to the wheels movement so that the vehicle slows down and, eventually, stops.

The steering of the vehicle is controlled using the *steering* variable, whose changes are smoothed to make vehicles turning progressively, thus becoming easier to control for users.

Therefore, the vehicle's pose and linear and angular speeds are controlled by a, f_b , and *steering* variables, whose values can be given by either by the ADS (all vehicles) or the HMI (only the ego vehicle). Indeed, all vehicles share the same kinematics model and control mechanism (figure 3, right) and have a simple ADS to be able to follow paths from the road graphs. The difference of ego vehicle with respect to the rest is that it also has an HMI for the user to interact with it (figure 3, top left), including the capability to define destinations spots. (The rest of vehicles are fully autonomous and select random destinations to move around.)



Fig. 3. All vehicles include a GPS and an ADS module to be able to follow paths to user-defined or random target positions, and only the ego vehicle needs an user interface (UI) module.

4.2 GPS system

The GPS module is a simplified version of an actual GPS of a vehicle that allows the driver to monitor its position and to specify destinations. For this, a 2D view of the world is obtained from a zenith camera on a flattened 3D world, which is a copy of the 3D scene where the heights of buildings are scaled down to prevent the taller ones from shadowing or hiding the roads. On top of this view, there is a position icon at the vehicle's current location and, eventually, another one at the destination point (Fig. 3, bottom left.)

When the user chooses a destination in the 2D map, this module computes the path in the road graph to reach it by using the A^* algorithm [31] that is available via the *AStar* class that comes with Godot's default library. (Unfortunately, the ID of the nodes for this class is 32-bit long while the OSM node IDs are 64-bit long, thus wrapper methods that make the conversion between 64-bit and 32-bit IDs have been added.)

Once a route has been calculated, a guiding arrow pointing towards the right direction is projected in the display when the car is driven in manual mode. In case of autonomous driving, the vehicle follows the path to the destination with eventual driver requests to hand over the control.

Vehicle User Control The HMI is designed to make the game usable with any computer, even without (gaming) steering wheels and pedals, so the user input recognizes WASD keys (arrow keys, too) as well as the space bar to steer, accelerate and brake the ego vehicle.

Note that the user control works seamlessly with the ADS component in order to allow the user to switch between manual and autonomous driving modes in a smooth manner at will, apart from those events that cause ADS requiring the driver taking control of the vehicle as soon as possible.

Vehicle ADS The ADS takes charge of assisting the driver to follow a predefined route or making the vehicle doing so in autonomous mode. To do so, it constantly moves the vehicle forward while steering it towards the next node position in the path. To make sharper turns, it brakes the vehicle when it is turning. It also keeps the vehicle at the right side of the road.

To follow a path, the ADS goes node to node and computes the vehicle orientation with respect to the target node in the current arc. Depending on the driving mode, it only informs the user about the direction to take or applies the right amount of steering directly. The *steering* is the *y* component of the resulting vector of the cross product between the forward vector \boldsymbol{P} of the vehicle and the target direction vector \boldsymbol{T} :

$$steering = (\mathbf{P} \times \mathbf{T})_y \tag{2}$$

Vectors P and T are computed from the normalization of the current linear velocity vector, and from the difference between the position of the next target node in the path the vehicle is following and the current position of the vehicle in 3D space.

Because node positions are the waypoints at the middle of the roads, they must be offset to the right to achieve that vehicles drive in the right side of roads. Therefore, for a given path to be followed, the position of all the nodes is shifted by the normal vector of each path segment. The length of this normal vector can be changed to determine how far from the right the vehicle will drive and could be configured to make it drive at a specific lane in multiple-lane roads. In this work, however, all roads are considered to have a single lane in both directions and the vehicle will drive at the center of the right lane.

4.3 Non-User-Controlled Vehicles

Apart from the ego vehicle, the game scene is populated with non-user-controlled vehicles at random spots. These vehicles simulate regular traffic on the roads and

create situations where user's attention and reaction speed can be assessed. As previously mentioned, they have the same model and components than the ego vehicle but for the user interface and the working modes of GPS and ADS.

In non-user-controlled vehicles, at the beginning of the simulation or at any time a destination is reached the ADS chooses a new random destination, asks the GPS for a path to it and starts following the new route.

The proximity sensors of these vehicles are used by the ADS's to brake or stop before an obstacle in the way. They are simulated by collision shapes in front of the vehicles that trigger events any time physical entities collide or stop colliding with them. For the sake of simplicity, vehicles fully stop when they detect the ego vehicle entering their collision shape to prevent from hitting it. For the rest of the entities, i.e. other vehicles or pedestrians, they slow down and eventually stop. In this work, we have not included this behavior in the ego vehicle, but it could be added to simulate an ADS with a collision avoidance system (CAS) with an autonomous emergency braking (AEB).

4.4 Pedestrians

Pedestrians are included in the simulated scene to make it more realistic and have an additional source of situations to test user reactions and behavior at the wheel. Like the vehicles, they are initially placed at random spots of the scene, at sidewalks. These spots are determined by offsetting node positions, which are at the middle of the roads, either to left or right side, using their widths.

Like the non-user-controlled vehicles, pedestrians move around randomly from their current node positions to a random neighbor node. Unlike them, they do not follow any path or have any specific destination, thus their behavior looks different to the users. They roam around their initial places so that their distribution in the environment is quite even, without forming clusters.

4.5 Chunk System

Populating the 3D world with vehicles and pedestrians may affect performance of the game when their number grows to the thousands. Fortunately, only those in the vicinity of the ego vehicle, at the possible sight of the player, are required at a given moment. Therefore, the game groups entities into chunks so that only those nearby the ego vehicle are simulated.

The *chunk system* is implemented as a dictionary so that chunk coordinates are the key to access a class object that holds the list of all the entities that are inside this chunk in addition to a variable that tells us whether this chunk is active or not. (Note that it could include any other entity besides vehicles and pedestrians.)

Chunks are rectangular cells of a grid that covers all the area of the 3D scene thus, entities belong to the chunk corresponding to their coordinates divided by the width and height of the cells. The cell size can be modified but should not be smaller than the size of a city block or too large so that near all entities are simulated at each simulation cycle. The activation or deactivation of a chunk depends on the distance from the user's coordinates. In the first case, all entities of the chunk are made visible to the user and their execution is resumed, while deactivation of chunks imply the opposite operations, namely, stopping the execution of the behaviors of the affected entities and their rendering too, thus saving CPU time and increasing performance of the simulation.

5 Experimentation

The software is publicly available through GitHub [https://github.com/bemo10/ Autonomous_Vehicle_Simulation] and runs on MS Windows platforms with the Godot engine (version 3.5 stable), and it can be modified freely. In the folder of the project there is a *README.md* file with information about how to select a map for the simulation and the user controls when running the *Autonomous Vehicle Simulation.exe* program.

By default, the simulated worlds are assumed to come from areas not larger than 50,000 OSM nodes and are populated with 10,000 pedestrians and 3,000 vehicles.

The performance of the simulation depends directly on the time to render the scene and, particularly, on the number of active entities at each cycle, which is related to the size of the chunks. For example, on a Windows 10 Home 64-bit computer with 16 Gb DDR4 1600 MHz RAM, an AMD Ryzen 5 3600 6-core CPU and a NVIDIA GeForce RTX 3060 Ti 8Gb GDDR6 GPU, the number of frames per second (FPS) range between 40 and above 100 for cases between 60 active entities down to 20.

5.1 User Interaction Assessment

The user attentiveness and reaction speed are measured when a potentially hazardous situation arises, and he or she must take over the control of the vehicle. These situations appear when the ego vehicle heads against another vehicle or a pedestrian. In the last case, note that, in addition to the regular behavior of walking at the sidewalks, pedestrians may eventually cross streets.

The game starts when the user chooses a destination, and the vehicle starts driving autonomously to that location. The driver must pay attention to other vehicles and to the pedestrian movements to regain control of the vehicle whenever any risky situation may take place. These cases are randomly fired at specific points of the journey and, for each case, the delay in taking over control and the outcome are recorded.

The game ends when the driver gets to the destination, and a report with the data of how has he or she managed each situation and a final score for that particular journey is displayed and, eventually, stored for further analyses.

As said, the game measures the time it takes to the driver to get the control of the vehicle from the moment a potential accident may occur until the moment that some manual input is done. Additionally, it scores the outcome of the situation, with positive values when nothing happens and negative ones otherwise. In the current version, the scoring system is simple: it adds $40/t_r$ points for each avoided accident (t_r is the reaction time) and subtracts 50 on car crashes or hitting pedestrians. (Note that, for the first cases, reaction times are also recorded.)

6 Discussion and Conclusion

Paradoxically, the more the technology provides us with safety while traveling in a vehicle, the easier drivers get distracted or pay less attention to driving. These phenomena are studied to develop technology that prevents their adverse effects from taking place. Notably, some studies conclude that one of the best strategies is drivers' education, which includes acquiring knowledge about ADAS and ADS and training.

Studies and onboard technology (ADAS, ADS, DMS, HMI, ...) development and validation require experimentation with drivers. While experimentation with actual vehicles and realistic simulators are still required, the only way to make large experiments and reach a broader audience is by using software than can be run in affordable platforms.

In this work, we prove that technology is mature enough to develop such type of software by using open-source tools such as OpenStreetMap and Godot game engine. The result simulation framework, though simple, can create a 3D world from a real region of interest taken, at its turn, from OSM, and of let users drive a PAV in it, with other autonomous vehicles and pedestrians moving around and causing situations that make the ego vehicle ADS requiring them to take over the control.

A basic scoring system is used to foster players to minimize their reaction times and to provide a simple example of how the framework can be used to train drivers to face these situations. Anyway, it can be hidden or modified to adjust it to the needs of any study or learning objectives, as well as the types and frequency of situations drivers' run into.

The software simulator reduces its computational power requirements by simulating only the objects that are in the chunks close to the ego vehicle and, therefore, it is possible to include other classes of objects and, particularly, traffic lights or signs. Additionally, the 3D scene can be enriched with non-essential details to get a better level of realism. This option, however, should be included without compromising simulation execution performance.

With this development, we believe that it is possible to use this kind of "light" simulation games to make studies with a larger number of participants and, though results might not be as accurate as with a more realistic platform, they would be more representative of the population. And, last but not least, they might be more useful as educational tools (e.g. in driving schools) and as complementary tools for psychometric tests.

Acknowledgments. This research has been partly funded by Catalan Government Research groups 2021SGR01623.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

- Kaur, P.; Taghavi, S.; Tian, Z.; Shi, W. "Survey on Simulators for Testing Self-Driving Cars." arXiv, 2021, doi: 10.48550/arXiv.2101.05337.
- Thorslund, B.; Thellman, S.; Nyberg, V.; Selander, H. "Simulator-based driving test prescreening as a complement to driver testing – Toward safer and more riskaware drivers." Accident, Analysis & Prevention, vol. 194, 2024, ISSN 0001-4575, doi: 10.1016/j.aap.2023.107335.
- Sánchez, J.S.; Ortiz, J.S.; Mayorga, O.A.; Sánchez, C.R.; Andaluz, G.M.; Bonilla, E.L.; Andaluz, V.H. "Virtual Simulator for the Taking and Evaluation of Psychometric Tests to Obtain a Driver's License." In: De Paolis, L.; Bourdot, P. (eds) Augmented Reality, Virtual Reality, and Computer Graphics. AVR 2019. Lecture Notes in Computer Science, vol 11613. Springer, Cham., 2019, doi: 10.1007/978-3-030-25965-5_11.
- Kaça, G.; İzmitligil, T.; Koyuncu, M.; Amado, S. "How well do the traffic psychological assessment systems predict on-road driving behaviour?" *Applied Cognitive Psychology*, 35(5), 1321–1337, 2021, doi: 10.1002/acp.3867.
- Strayer, D.L.; Watson, J.M.; Drews, F.A. "Cognitive distraction while multitasking in the automobile." Chapter 2 of Advances in Research and Theory, Psychology of Learning and Motivation, vol 54, Elsevier 2011, doi: 10.1016/B978-0-12-385527-5.00002-4.
- Zangi, N.; Srour-Zreik, R. Ridel, D.; Chassidim, H.; Borowsky, A. "Driver distraction and its effects on partially automated driving performance: A driving simulator study among young-experienced drivers." *Accident Analysis & Prevention*, vol. 166, 2022, doi: 10.1016/j.aap.2022.106565.
- Dengbo, H.; Dina, K.; Birsen, D. "Distracted when Using Driving Automation: A Quantile Regression Analysis of Driver Glances Considering the Effects of Road Alignment and Driving Experience." *Frontiers in Future Transportation*, vol. 3, 2022, doi: 10.3389/ffutr.2022.772910.
- Lambert, F. "Tesla Autopilot results in decreased driver attention, new study finds." Electrek. Sep. 13, 2021. Available online: https://electrek.co/2021/09/13/teslaautopilot-decreases-driver-attention-new-study/.
- 9. Regan, M.A.; Hallett, C. "Driver Distraction." Handbook of Traffic Psychology, 2011.
- 10. "Driver distraction 2015." Report by the European Road Safety Observatory, 2015.
- Chen, W.; Liu, W. "HMI Design for Autonomous Cars: Investigating on Driver's Attention Distribution." In: Krömker, H. (eds) *HCI in Mobility, Transport, and Automotive Systems. HCII 2019. Lecture Notes in Computer Science*, vol 11596. Springer, Cham. 2019, doi: 10.1007/978-3-030-22666-4 7.
- Papantoniou, P.; Papadimitriou, E.; Yannis, G. "Assessment of Driving Simulator Studies on Driver Distraction." Advances in Transportation Studies, vol. 35, 2015, pp 129-144.
- Fernández, A.; Usamentiaga, R.; Carús, J.L.; Casado, R. "Driver Distraction Using Visual-Based Sensors and Algorithms," *Sensors*, 16, 1805, 2016, doi:10.3390/s16111805.
- Jin, L.; Niu, Q.; Hou, H.; Xian, H.; Wang, Y.; Shi, D. "Driver Cognitive Distraction Detection Using Driving Performance Measures." *Discrete Dynamics in Nature and Society*, Hindawi Publishing Corp., 2012, doi: 10.1155/2012/432634.

- Ahangari, S.; Jeihani, M.; Rahman, M.M.; Dehzangi, A. "Predicting driving distraction patterns in different road classes using a support vector machine." *International Journal for Traffic and Transport Engineering*, 2021, doi: 10.7708/ijtte.2021.11(1).06.
- Srinivasan, K.; Garg, L.; Datta, D.; Alaboudi, A.A.; Jhanjhi, N.Z.; Agarwal, R.; and Thomas, A.G. "Performance comparison of deep CNN models for detecting driver's distraction." *Computers, Materials & Continua*, vol. 68, no.3, pp. 4109–4124, 2021, doi: 10.32604/cmc.2021.016736.
- IIHS creates safeguard ratings for partial automation. Insurance Institute for Highway Safety (IIHS), Jan. 20, 2022 [https://www.iihs.org/news/detail/iihs-creates-safeguard-ratings-for-partial-automation].
- Karatas, N.; Yoshikawa, S.; Tamura, S.; Otaki, S.; Funayama, R.; and Okada, M. "Sociable driving agents to maintain driver's attention in autonomous driving." 2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), 2017, pp. 143-149, doi: 10.1109/ROMAN.2017.8172293.
- Perrone, A. "How Automakers Plan to Keep Drivers Attentive in Self-Driving Cars." *Endurance*, March 08, 2019 [https://www.endurancewarranty.com/learningcenter/tech/keep-driver-attention-self-driving-cars/].
- Cunningham, M.L.; and Regan, M.A. "Driver distraction and inattention in the realm of automated driving." *IET Intell. Transp. Syst.*, 12: 407–413. 2018, doi: 10.1049/iet-its.2017.0232.
- Beckers, N.; Siebert, L.C.; Bruijnes, M.; Jonker, C.; and Abbink, D. "Drivers of partially automated vehicles are blamed for crashes that they cannot reasonably avoid." *Scientific Reports*, vol. 12(1), 2022, doi: 10.1038/s41598-022-19876-0.
- Hungund, A.P. "Systematic Review of Driver Distraction in the Context of Advanced Driver Assistance Systems (ADAS) & Automated Driving Systems (ADS)." Masters' Theses, 1243. 2022, doi: 10.7275/30956149.
- Miller, D.; Sun, A.; Johns, M.; Ive, H.; Sirkin, D.; Aich, D.; and Ju, W. "Distraction Becomes Engagement in Automated Driving." *Proceedings of the Human Factors* and Ergonomics Society Annual Meeting, 59(1):1676-1680, September 2015, doi: 10.1177/1541931215591362.
- Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. Report J3016_202104. Society of Automotive Engineers (SAE), 2021. [https://www.sae.org/standards/content/j3016202104/]
- Linietsky, J.; Manzur A.; and the Godot community. "Godot Version 3.5 documentation" [https://docs.godotengine.org/en/stable/index.html]
- Dosovitskiy, A.; Ros, G.; Codevilla, F.; López, A.; and Koltun, V. "An Open Urban Driving Simulator," *Proceedings of the 1st Annual Conference on Robot Learning*, 1-16, 2017 [https://carla.org/]
- 27. The OpenStreetMap wiki [https://wiki.openstreetmap.org/wiki/Main_Page].
- 28. An API for read-only queries of OSM data [http://overpass-api.de/].
- 29. 3D layer of buildings [https://osmbuildings.org/].
- 30. scailman (user id). Low-Poly Small Car. Sketchfab. 2019. [https://skfb.ly/6QZtA].
- Hart P.; Nilsson N.; and Raphael B. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics* 4(2):100–7, 1968, doi: 10.1109/tssc.1968.300136.

Analysis of strategies to apply federated learning in the field of robotics

Roi Martínez Enríquez¹, Nicolás Fraga Corredoira², José Miguel Burés Amatriaín¹, Roberto Iglesias Rodríguez¹^[0000-0002-6279-5190], Francisco Javier García Polo²^[0000-0002-5638-5240], and Xosé Ramón Fernández Vidal¹^[0000-0001-9388-7461]

¹ CiTiUS - Universidade de Santiago de Compostela, Santiago de Compostela, Spain https://citius.gal/

² EPSE - Universidade de Santiago de Compostela, Lugo, Spain

Abstract. In this paper we analyse the use of techniques that will help us to apply federated learning in the context of robotics. Federated learning allows the building of a global model from data that is collected locally, at different devices and without data sharing. Nowadays there are already solutions that face the problem of data heterogeneity (non i.i.d). Nevertheless, even when there seem to be a common task that guides the federated learning, the local implementation or details are not exactly identical. This is particularly important in the case of robotics. In this paper we analyse the performance of different techniques that allow us to identify malicious learners, or minority dissenting learners. This identification will allow to modulate the impact of these learners in the global model. Finally, in order to apply federated learning in the context of robotics, we also need strategies which allow the learning of a model when the sensors on the robots are heterogeneous.

Keywords: Federated Learning \cdot Malicious attacks \cdot Dissenting learners \cdot Robotics

1 Introduction

In the last years, regulations on privacy and security of personal data have had a significant impact on machine learning methods based on data centralization. These regulations have led to an increase in requirements, as well as the costs needed to satisfy them. Although there are limitations to data centralization, the training of deep models still requires large amounts of data. In this context, Federated Learning (FL) has emerged as a strategy able to collaboratively achieve big models but preserving data privacy.

FL is a machine learning approach that allows the achievement of a global model on a central server, thanks to the contribution of a set of local learners (clients). The data and training process remain on local devices (clients). A significant number of FL algorithms follow a similar process to achieve this goal: A current copy of the global model is sent from the server to the clients. Then, each client updates its local version of the global model using its own local data, and sends the local gradients of the weights of the global model back to the server. After this, the central server updates the global model by aggregating the local gradients received from the clients. These three steps are repeated until the global model converges.

Nevertheless, FL has yet important open challenges and it has not been seriously applied in the context of robotics, due in part to the fact that the standard FL approach imposes constraints that are unrealistic in the context of robotics. Some of them are related with the diversity of hardware and skills since robots operate in increasingly complex and time-varying environments coworking with people. Since robots are endowed with different sensors it might be difficult to achieve a model valid for them all. On the other hand, there will be highly non IID data (independent and identically distributed) collected locally by the learning clients, due to the heterogeneous environmental and operational conditions, as well as the robot's characteristics.

To illustrate, consider the example of a domestic cleaning robot, which uses obstacle avoidance behaviour to navigate and clean the ground. The following is a simple but illustrative example related to the content of this paper: Imagine a scenario in which four robots (clients) are situated in four different homes and each robot has a different strategy for obstacle avoidance. The first robot always turns left to avoid the obstacle as soon as it detects it. The second robot always turns right when it encounters an obstacle. The third robot takes more risks and waits until it is closer to the obstacle to turn both right and left. Finally the fourth robot is more cautious and avoids obstacles at a greater distance by also turning both right and left. If we use FL in this example and attempt to find a global model that works properly for all four clients, it is likely that the model will not converge, resulting in a quite disastrous behaviour in which each robot fails to avoid obstacles and directly crashes into them.

In the next sections we will present a strategy aimed to achieve a global model that can cope with sensor heterogeneity. On the other hand we will also analyse different techniques aimed to detect and reflect either malicious attacks, or client's behaviour that differ from the majority, and which might compromise the global model.

2 Methodology

2.1 Achieving global models in presence of sensor heterogeneity

Standard FL imposes a model that is common for all clients. When this model is an artificial neural network, a deep network for example, this restriction means that the network's topology is the same for all clients. As we pointed out in the introduction, there might be differences in robot's sensors, therefore the sensor information (both dimension and characteristics) might vary amongst the robots even when solving the same task. As a first approach to sort this problem out, we will analyse the use of a strategy that comes from the transfer learning scenario. Most commonly in transfer learning, when convolutional neural networks are used (CNN), convolutional layers, especially the first ones, tend to freeze because they learn low-level features (edges, textures, simple patterns) that are generally applicable to many different tasks. These basic features do not change much from task to task. However, in our case we will do the opposite, i.e., only the fully connected layers will be shared during federated learning, while the convolutional layers will be locally adapted, so that they can be personalized for every client. In this case, all clients must agree in the embeddings. This means that similar environment should correspond with close embeddings for all robots. The CNN will reduce the dimensionality, extracting only the important features (embeddings coming out of the convolutional layers). Then, the data is passed through the fully connected layers to make a decision for the final output. These embedding/latent vectors should be common for all robots.

2.2 Detecting discrepancy or malicious attacks

Considering the simple exampled pointed out in the introduction, "obstacle avoidance", we could see that although the task "seems to be" the same for all learners, there might be important differences in the strategies applied when data is collected locally for supervised training. As it is pointed out in [6], the problem is to determine when the federation, i.e. collaboration, will really help. If the user's distribution does not deviate too much from other user's data distribution, then collaboration might be beneficial. But when the local distribution is far from being a representative sample of the overall distribution, or even the task/problem being solved is different, it is too difficult to find a global model that is good for all clients, and thereby independent models (one-per-device) are preferable. The problem is knowing the task similarity amongst learners beforehand. Usually, federated learning, and in particular federated averaging, weights the contribution of each learner in the global model considering only the normalized percentage of data in each learner. Nevertheless, in our case, we plan to work with what is called personalized federated learning. There will be a global model, but also local/adapted versions of this model in every client. As we just pointed out, it might happen that there is a minority of learners whose local data is very non i.i.d or heterogeneous, while an important ratio of learners agree in the characteristics of the task being learnt, or have highly correlated local datasets. Because of this, we introduce a second group of weights that will reduce the influence of the few dissenting learners in the global model, while on the contrary increases the relevance of the learners that agree amongst them, when building the global model. To achieve these new set of weights that determine the importance of every learner in the federation, we need techniques able to build these weights detecting malicious or suspicious behaviours that disagree with the majority. We have analysed the use of three alternatives which can be useful for our purpose. The idea is to use a soft decision regime that aggregates model updates weighted by their trust scores, or suspicious scores.

FLDetector We decided to apply part of the technique described in the papers published by Zhang et al. to detect malicious clients (FLDetector [4], and FedRecovery [5]), to detect when a minority of the learners are clearly dissenting from the rest. If we call w_t the weights corresponding to the current global model, and g_i the gradient of the global model computed by the *i* client: $g_i = \nabla f(D_i, w)$, we can estimate the next update (gradient), using the integrated Hessian for that specific client:

$$g_i^t = g_i^{t-1} + H_i^T(w_t - w_{t-1})$$

where H_i^T is the integrated Hessian for client *i* in interaction *t*. However, instead of working with the estimated Hessian for client *i*, we work with \hat{H}^t , in instant *t*, an estimated global Hessian which is the same for all clients.

$$\hat{g}_{i}^{t} \approx g_{i}^{t-1} + \hat{H}^{T}(w_{t} - w_{t-1}) \tag{1}$$

We use the L-BFGS algorithm to estimate the Hessian Vector Product $H\Delta w$. Therefore, using (Eq. 1), we can estimate which one should be the next gradient for every client. Thus, comparing the estimated gradient with the observed one, we can detect when a client dissents from the majority. Thus, if we denote by d^t the Euclidean distance amongst observed and predicted gradients:

$$d^{t} = [\|\hat{g}_{1}^{t} - g_{1}^{t}\|_{2}, \|\hat{g}_{2}^{t} - g_{2}^{t}\|_{2}, ..., \|\hat{g}_{n}^{t} - g_{n}^{t}\|_{2},]$$

we normalize the vector $\hat{d}^t = d^t/|d^t|_1$, and then we can compute a suspicious score for every client, s_i^t , as the average normalized Euclidean distance in the past N interactions: $s_i^t = \frac{1}{N} \sum_{r=0}^{N-1} \hat{d}_i^{t-r}$

Therefore, this scoring system is based in the evaluation of consistency between the predicted model update, using L-BFGS, and the actual update, using FedAvg. A client's update exhibiting a substantial deviation from expected gradients, quantified through these difference scores, signals a higher likelihood of malicious behaviour or local data difference with respect to the majority.

Flare This strategy [1] builds trust scores, comparing the outputs of the penultimate layer of the models that have been updated by the clients. Basically, the server sends the global model to the clients. Then, each local client trains the copy of the global model with the local data and sends the model update back to the server. The server uses this model updates to compute the trust scores. To do so, it uses a very small set of auxiliary data. The server feeds these auxiliary data to the models returned by the clients, and gets the PLR (embedding provided by the penultimate layer). Then, the server measures the distance (i.e. the maximum mean discrepancy, MMD) between the PLRs of any two models to estimate the trust score. Under the assumption that suspicious clients are fewer than honest ones, FLARE assigns a score to each model update based on the pairwise PLRs diecrepancies amongst all model updates. Let's assume that $R_j := \{g_{w_j}(x_1), ..., g_{w_j}(x_m)\}$ represents the PLRs provided by the model returned by client j, for every pattern x_i of the auxiliary data at the server. Then it is possible to compute the MMD between any two sequences R_i and R_j :

$$MMD(R_i, R_j) = \frac{1}{m(m-1)} \left[\sum_{a \in R_i} \sum_{b \in R_i, b \neq a} k(a, b) + \cdots \right]$$
$$\cdots \sum_{a \in R_j} \sum_{b \in R_i, b \neq a} k(a, b) - 2 \sum_{a \in R_i} \sum_{b \in R_j} k(a, b) \right]$$

where k(.) is a Gaussian Kernel function. Flare selects the top 50% nearest neighbors for each local model based on the MMD scores. The count ct_i for client i increases by one, once i is selected by any client j, with $j \neq i$. The count ct_i indicates a trustworthiness. Thus, the trust score is computed as:

$$S_i = \frac{exp(ct_i/\tau)}{\sum_{k=1}^n exp(ct_k/\tau)}$$

where τ is the temperature parameter, and S_i is in the interval [0,1].

FLTrust Like in FLARE, FLTrust [2] will compute trust scores for every client in the federation, using a clean dataset stored at the server, called *root dataset*. The cyclical process is the same as always. The server sends the global model to the clients, which update their copies and send them back to the server (model updates). In this case, the server also trains the global model (which has been sent to the clients), using the *root dataset*. FL trust considers both, the server model update and the client's model updates to compute the trust indexes. Basically, it takes the directions of the model update based on its direction similarity with the server model update. The *cosine* similarity is used to compute the angle between the weight vectors. However, the cosine similarity alone faces a challenge, if a local update and the server local update are in the opposite directions their cosine similarity will be negative. To avoid these negative values FLTrust clips the consine similarity using the ReLU operation for clipping. Thus, the trust score (TS_i) , for every client *i*, is defined as:

$$TS_i = ReLU(\frac{\langle gi, g_0 \rangle}{||g_i||\dot{|}g_j||})$$

where g_i represents the *i* client's model update, g_0 the server model update and, as usual, ReLU(x) = x if x > 0, and 0 otherwise.

It is important to be aware that the indexes for FLTrust and Flare reflect confidence, the higher the value of the index for each client the more reliable that client is. On the contrary, FLDetector computes suspicious indixes, the higher the index for a client the more suspicious it is.

3 Experimental Results

In this section we present the performance of the strategies described in the previous section to achieve shareable models, or to detect anomalies (trust and suspicious scores).

3.1 Datasets

In our study, we employ four datasets.

MNIST Dataset Classic dataset used widely in machine learning for digit recognition tasks. It consists of 70,000 grayscale images of handwritten digits, split into 60,000 training and 10,000 testing samples.

DigitFive Collection: Aggregated compilation specifically designed to challenge federated learning systems with a realistic scenario of data heterogeneity. It consists of the following five distinct digit datasets: MNIST (described before). MNIST-M: Features a blend of MNIST digits superimposed on patches randomly extracted from color photos from BSDS500 dataset. This results in 59,000 color images that combine the simplicity of MNIST digits with complex color backgrounds, enhancing the challenge for digit recognition models. SVHN (Street View House Numbers): Comprises over 63000 real-world, color images of house numbers taken from Google Street View images. This dataset introduces complex backgrounds and varying digit styles, simulating more challenging and diverse visual recognition tasks. USPS (United States Postal Service): Contains 9,000 grayscale images derived from scanned envelopes by the USPS, which are normalized and centered. This dataset provides a variation in handwriting and digit arrangement. Synthetic Digits (SYN): Includes 12,000 synthetic images of digits generated with random variations in fonts, colors, and backgrounds.

Turtlebot Collection: Obstacle Avoidance Collection The third collection of datasets, named *Turtlebot collection*, consists of data collected by a Turtlebot IV robot using an obstacle avoidance behavior. In this case we built a collection of 4 datasets, the inputs are always the same (information provided by a laser scanner), but as we will describe next the behaviour of the robot is different for each data set of the collection. For each one of the 4 datasets of the collection, the data was collected over an approximate period of 6 minutes in 4 different environments, each with different characteristics, in order to have a heterogeneous environments. In total, around 7500 different examples were collected, with approximately 1500 examples per environment. The data collected in each environment consists of the inputs patterns (X), which are composed of the vectors with the 1440 measurements of the LiDAR sensor, and the output labels (Y), which are the linear and angular velocity in the X and Z axis of the robot, respectively.

Dataset 1 in the collection: Threshold 0.55 In this case the robot turns whenever detects an obstacle below a threshold distance of 0.55 m. When the robot turns it can do it either to the right or the left. In this case to determine the direction of turning we used repulsive forces.

 $Dataset\ 2$ in the collection: Right The threshold distance is the same as before, 0.55 m, but in this case the robot is only allowed to turn right when an obstacle is detected.

Dataset 3 in the collection: Left Once again the threshold distance is the same as the two previous cases, 0.55m, but in this case the robot is only allowed to turn left when an obstacle is detected.

Dataset 4 in the collection: Threshold 0.7 In this case we change the threshold distance below which the robot will turn when an obstacle is detected, the new threshold is 0.70 m. Like in the first case, the robot will be allowed to turn either right or left using the repulsive forces described before.

Waffle Collection This new collection, consists of data collected by a Waffle robot using the obstacle avoidance behaviour explained for the Turtlebot robot in the previous section. For the Waffle we just collected one dataset, with the threshold set to 0.50 m and allowing the robot to turn both right and left when detecting an obstacle. The data collected in each environment consists of the input patterns (X), composed of the 231 measurements of the LiDAR sensor, and the output labels (Y), composed of the linear velocity in the X-axis of the robot and the angular velocity in the Z-axis of the robot. In total, around 5000 samples.

4 Analysis of poisoning attacks

We focus on evaluating the resilience of federated learning models against adversarial threats by implementing two distinct types of attacks on the MNIST dataset: the backdoor attack and the mean attack. Targeted attacks, (e.g. Backdoor [3]), are designed to subtly embed malicious behavior in a model, which is activated under specific conditions without affecting overall performance. The backdoor attack in our experiments is implemented by training the compromised model with images that contain a specific pattern associated with a target label. This is designed to induce the model to mistakenly associate the presence of this pattern with the target label, effectively embedding a backdoor in the model. The attack is further enhanced by a scaling attack strategy, where the gradients sent by the malicious client to the global model are scaled up. This escalation in gradient values aims to increase the influence of the malicious client on the global model, making the backdoor more effective during the model aggregation phase. The second attack we have considered is the *Mean (untargeted)* attack. In this case it aims to broadly degrade model performance, challenging its overall accuracy and effectiveness. The mean attack involves manipulating the direction of the gradients sent by the compromised client to the global model. Specifically, the gradients are inverted before being sent. This inversion is intended to disrupt the learning process by pushing the global model away from the optimal solution, rather than towards it.

To carry out the FL with MNIST, we worked with ten clients. In the initial scenario only benign clients featured, providing a baseline by assessing their aggregation algorithm, based on trust scores, under optimal conditions. Subsequently, the poisoning attacks were executed, where three of the ten clients compromised the training process testing the system's resilience against intentional data manipulations. In alignment with the methodologies outlined in Wang et. al.[?], global model aggregation occurs every five local training epochs. FLTrust adjusts its aggregation schedule to ensure an equivalent amount of training, aligning with FLARE's schedule for a consistent comparison of different techniques. Specifically, while FLARE performs global aggregation every five local training epochs, FLTrust performs global aggregation at every local epoch. To achieve an equivalent amount of training, if FLARE has 20 global aggregations, FLTrust conducts 100 global epochs.

Both FLARE and FLTrust methodologies necessitate the presence of a dataset on the global server to function effectively. FLARE, as previously discussed, utilizes a small dataset on the server to monitor the behavior of each participant's PLR within the latent space. For all the experiments conducted, this dataset consisted of synthetically and randomly generated data, to ensure that the dataset did not interfere with the scoring calculations, thereby providing an unbiased baseline to assess the integrity of data submitted by clients. In contrast, for FLTrust, the dataset on the server acts as a trust indicator and thus, its content holds substantial importance. In both methodologies, the server model employs a small set of benign data for training—using data without attacks in the case of MNIST.

During the implementation of FLTrust, it was observed that the relatively small dataset on the server resulted in a slower learning rate for the server's local model. This fact not only affected the server's learning efficiency but also had a direct impact on the global model by adapting all updates to the slower norm of the server's local update. To remedy this issue, the dataset selected for the server was replicated to match the size of the datasets used by the other clients, ensuring a more balanced and effective learning rate across the network, although an alternative solution may have been increasing the number of local epochs exclusively for the server to compensate for the smaller dataset size.

MNIST Dataset Figures 1, 2 and 3 show the results of applying FLdetector, Flare and FLTurst to detect malicious clients. As we can see in these figures, the three methods demonstrate remarkable success in detecting malicious clients. Despite of the good performance of all methods, it is interesting to notice that in scenarios without attacks, FLARE shows an important variance in the scores. This variation reflects the method's sensitivity to the relative positioning of client data. On the contrary, FLTrust, which anchors its trust assessments in the server-held data, and FLDetector, which forecasts the global update using the historical information provided by each client, exhibit a more stable set of indexes, i.e. a much more uniform distribution of influence.



Fig. 1: Average Suspicious Scores per Client Across Different Attack Scenarios in FLDetector. Note that a higher score indicates a higher level of suspicion. Consequently, clients with higher scores are considered less trustworthy and are given lesser weight in the aggregation process, or might even be excluded altogether, following the original methodology proposed by Fang et. al [?]



Fig. 2: Average Trust Scores per Client Across Different Attack Scenarios in FLTrust



Fig. 3: Average Trust Scores per Client Across Different Attack Scenarios in FLARE

5 Detecting dissenting or suspect minorities

With this case we will analyse the performance of FLTrust, Flare and FLDetector. When only a small percentaje (minority) of learners dissent from the rest of the learners. We will work with two collections: DigitFive and the Turtlebot data. In this case, we work with five learners, but in all the experiments four of them used the same dataset from the collection, while one used a distinct one.

Again, let us note that both FLARE and FLTrust methodologies necessitate the presence of a dataset on the global server to function effectively. In both cases, the server model employs a small set from the predominant dataset.

DigitFive Dataset Table 1 displays the average trust scores across multiple runs for FLTrust, FLARE and FLDetector, respectively, including the mean scores for clients sharing the same dataset, labeled as *Com*, alongside scores for the distinct client. As we can see in this table, the three methods are able to properly detect the dissenting client. It is noteworthy that FLTrust and FLDetector consistently assign the expected scores (higher in FLTrust and lower in FLDetector) to the majority clients in all tested combinations. In contrast, FLARE assigns incorrect scores in scenarios where the SYN dataset predominates, among others. Furthermore, it is observed that in situations where all clients possess the same data, the scores assigned by FLTrust and FLDetector are significantly more stable than those calculated by FLARE (something that we had seen in the previous section too).

Finally, a notable distinction is observed between the methods regarding how dataset characteristics influence trust scores. In FLDetector, and specially in FLTrust, the difference in trust scores clearly aligns with the nature of the datasets being handled. For instance, in FLTrust, MNIST scores are more closely aligned with SYN and MNIST, rather than with SVHN, illustrating the methods' sensitivity to dataset characteristics. Conversely, FLARE appears to be more arbitrary in its scoring, basing assessments exclusively on proximity relations, which may lead to less predictable results. These observations reinforce earlier discussions from the MNIST dataset, highlighting the importance of a reliable dataset stored on the server which aids in stabilizing the method.

Turtlebot Dataset In this case, Tables 2, 3 and 4 show that, once again, FLTrust, FLDetector and FLARE are able to detect the dissenting client. Regarding the data held at the server in the case of FLTrust, we analysed two different approaches: the first approach involves distributing the total data from the majority class among the benign clients and the server, thereby treating the server as another benign participant with equivalent access to training data. This method aims to enhance the overall homogeneity of the data being processed, potentially stabilizing the learning process across the network. The second approach, whereby 20% of the training data from each client is allocated to the server, tests the behavior of FLTrust under conditions where obtaining a clean
					Inc	lepend	lent clie	ent			
FLDet.	-	MN	IST	MNI	ST-M	SV	HN	S	ΎΝ	US	PS
	-	Dif	Com	Dif	Com	Dif	Com	Dif	Com	Dif	Com
Common Clients	MNIST MNIST-M SVHN SYH USPS	.200 .287 .298 .304 .335	.200 .178 .175 .174 .166	.315 .201 .248 .254 .346	.171 .200 .188 .186 .163	.336 .259 .200 .263 .534	.166 .185 .200 .184 .117	.298 .264 .268 .198 .326	.175 .184 .183 .200 .168	.272 .277 .308 .290 .207	.182 .181 .173 .178 .198

		Independent client											
FLTrust		MN	IST	MNI	ST-M	SV	HN	S	ζN	US	SPS		
		Dif	Com	Dif	Com	Dif	Com	Dif	Com	Dif	Com		
Common Clients	MNIST MNIST-M SVHN SYH USPS	.797 .611 .295 .309 .823	.802 .875 .919 .741 .985	.493 .899 .564 .446 .246	.834 .900 .894 .873 .780	.172 .319 .950 .212 .759	.841 .870 .945 .788 .982	.278 .508 .543 .793 .354	.845 .912 .939 .815 .714	.533 .502 .336 .407 .626	.772 .860 .918 .833 .737		

FLARE		Independent client											
		MNIST		MNIST-M		SVHN		SYN		USPS			
	-	Dif	Com										
Common Clients	MNIST MNIST-M SVHN SYH USPS	.196 .204 .069 .218 .037	.201 .199 .233 .196 .241	.034 .201 .065 .091 .090	.242 .200 .234 .227 .227	.021 .052 .155 .240 .041	.245 .237 .211 .190 .240	.078 .165 .043 .395 .035	.531 .209 .239 .151 .241	.081 .113 .028 .242 .485	.230 .222 .243 .190 .129		

 Table 1: Comparative averages of FLDetector, FLTrust and FLARE scores across

 DigitFive dataset configurations for five clients, with four sharing the same dataset

 and one independent client in varying scenarios.

and sufficient dataset for the server training is challenging. This method deliberately takes a representative sample from each participating client's data, despite the inherent risk that not all data may be from a uniform dataset. By doing so, it seeks to enhance the server's comprehensive understanding of the data variations across the network. This strategy is designed not only to test the robustness of FLTrust under less ideal conditions but also to explore how well the method can manage and integrate diverse and potentially inconsistent data inputs to guide the aggregation process effectively.

		Independent client									
FLTrust Ap Common Th Clients Rig Le	Approach 1	Thresh	old 0.55	Thresh	old 0.7	Ri	ght	Le	eft		
		Dif	Com	Dif	Com	Dif	Com	Dif	Com		
Common Clients	Threshold 0.55 Threshold 0.7 Right Left	.575 .281 .108 .104	.636 .341 .360 .290	.299 .322 .146 .042	.482 .354 .379 .291	.133 .290 .437 .011	.487 .410 .374 .323	.050 .084 .004 .430	.412 .289 .362 .309		

		Independent client									
FLTrust	Approach 2	Thresh	old 0.55	Thresh	old 0.7	Ri	ght	Le	eft		
		Dif	Com	Dif	Com	Dif	Com	Dif	Com		
Common Clients	Threshold 0.55 Threshold 0.7 Right Left	.477 .547 .381 .170	.489 .505 .407 .433	.361 .481 .233 .098	.537 .428 .462 .428	.237 .307 .662 .177	.525 .370 .498 .259	.230 .267 .090 .386	.495 .274 .298 .487		

Table 2: Comparative averages of FLTrust confidence scores across TurtleBot dataset configurations for five clients, with four sharing the same dataset and one independent client in varying scenarios. Server in approach 1 receives data as another benign participant, while in approach 2 it is trained with 20% data of each client.

The results of both FLTrust approaches are shown in Table 2. As it is intuitively expected, we perceive a better performance of FLTrust in the first scenario.

On the other hand, FLARE does not require additional perspectives for server data handling, as the data used in the server are again synthetically and randomly generated. The scoring differential in FLARE is much more pronounced compared to FLTrust in scenarios where both datasets are quite different (*Left* or *Right* datasets are used). This outcome highlights FLARE's reliance on comparing updates between clients to assess trustworthiness.

Lastly, as shown in Table 4, FLDetector exhibits scoring patterns similar to those of FLTrust, contrasting with the more pronounced scoring differentials

		Different client										
FLARE		Thresh	old 0.55	Thresh	old 0.7	Ri	ght	Le	eft			
		Dif	Com	Dif	Com	Dif	Com	Dif	Com			
Common Clients	Threshold 0.55 Threshold 0.7 Right Left	.369 .181 .011 .011	.158 .205 .247 .247	.149 .450 .011 .011	.213 .137 .247 .247	.111 .032 .524 .011	.222 .242 .119 .247	.016 .024 .011 .336	.246 .244 .247 .166			

 Table 3: Comparative averages of FLARE confidence scores across TurtleBot dataset configurations for five clients, with four sharing the same dataset and one independent client in varying scenarios

observed in FLARE. Both methods assign confidence scores based on the level of data disparity among the clients, effectively adjusting their evaluations to the degree of dataset differences. For instance, clients which only differ in the distance threshold settings of the robot, receive much more similar scores than the other possible combinations. FLTrust and FLDetector tend to offer more stable and consistent scoring when data characteristics are similar.

6 Achieving a sharable model

As we pointed out in section 2.1, we have to deal with the fact that sensors on the robot might have different characteristics and position. In this case we work with two different robots: Turtlebot IV and Waffle. The laser scanner of the Turtlebot IV provides 1440 readings, while in the case of the Waffle it only provides 231. In both cases we invert the sensor readings and normalize the vectors (standard normalization). Nevertheless, in the Turtlebot IV we get infinite values when a reading is out of range or when there is an error taking the measurement. To deal with these infinite values, we analyse the previous adjacent value in the vector of readings: if the previous adjacent value is lower than 0.15 we replace the infinite value with 0.15, if the previous adjacent value is higher than 15 we replace the infinite value with 30. If neither of the two options are fulfilled we replace the infinitive value for the previous adjacent value. Nevertheless, in the case of the Waffle the problem is different. The vector of sensor readings not always has the same size (when it can not acquire all readings it returns a vector without some components). In this case we fixed the size of the vector readings to 231, therefore when we get a vector with less sensor readings we replicate the last one. If the vector has more than 231 we discard the last values. Therefore, we see that in the case of the Turtlebot IV and Waffle, despite they are similar robots, not only the dimension of the sensor readings (input to our network) is very different, but also the artifacts (infinite or wrong acquisitions).

We created two models from the Turtlebot IV dataset (model_1_Turtle and model_2_Turtle).

FLD-tt		Different client										
FLDetector		Thres	hold 0.	55 Thres	hold 0.7	Ri	ght	Left				
		Dif	Com	Dif	Com	Dif	Com	Dif	Com			
Common Clients	Threshold 0.55 Threshold 0.7 Right Left	.184 .167 .242 .249	.204 .208 .189 .188	.230 .180 .242 .240	.193 .205 .189 .190	.246 .300 .168 .270	.188 .175 .208 .182	.322 .268 .354 .173	.170 .183 .162 .207			

Table 4: Comparative averages of FLDetector confidence scores across TurtleBot dataset configurations for five clients, with four sharing the same dataset and one independent client in varying scenarios

model_1_Turtle The CNN implemented consist of one convolutional-1d layer, MaxPool-1d layer, convolutional-1d layer, MaxPool-1d layer, followed by 4 FC layers of 64, 14, 12 and 2 neurons.

 $model_2$ Turtle This second model consists of two MaxPool-1d layers (the kernel size = 4), followed by 4 FC layers of 64,18,12 and 2 neurones each.

Once these models were trained, we adapted them for the Waffle robot. In this case, we changed the convolutional layers (to face the new readings), but we froze the weights of the fully connected layers, which are a copy of the Turtlebot models.

model_1_Waffle In this case we adapted model_1_Turtle for the Waffle robot. We worked with a model consisting of one convolutional-1d layer (kernel size=5), MaxPool layer (kernel_size=3), convolutional layer (kernel size =3), and Max-Pool layer (kernel_size=2). The fully connected layers have a copy of the weights of model_1_Turtle and are frozen during the learning process, and therefore the topology of these layers are the same as before (64,14,12,2).

model_2_Waffle In this case we adapted model_2_Turtle for the Waffle robot. We worked with a model consisting of one convolucional-1d layer (kernel_size=1), another convolutional-1d layer (kernel_size=3), a MaxPool layer (kernel_size=3), and a linear layer of 90 neurones. The fully connected layers have a copy of the weights of model 2 Turtle and are frozen during the learning process.

In both cases, model_1_Waffle and model_2_Waffle converged and exhibit a low error in the test set (20% of the data was saved for test). In this article what we have implemented to reach sharable models is really transfer learning. We have seen how we can adapt the model of one robot to the other robot (freezing the fully connected layers). Nevertheless, as part of the future work we will work with a federated learning in which both models (the one in the Turtlebot and the one in the Waffle) are being learnt simultaneously. In this case, the federated learning will be applied on the weights of the fully connected layers, while the convolutional layers will be modified only locally.

7 Conclusions

If we want to apply federated learning in the context of robotics, there are at least two challenges that must be faced: On one hand, we must reach sharable models, due to the fact that the sensors on the robots might differ amongst them. On the other hand, even if the task is the same, most probably local data is heterogeneous due to the hardware of the robot, the way data is collected, etc. Due to this, we will probably need personalized federated learning, which will allow to achieve local models that are close but not identical to the global model. In this case, the use of strategies that allow the detection of malicious or dissenting clients will help to reduce their impact on the global model. We have compared three strategies: FLARE, FLDetector and FLTrust. All of them provide a set of indexes which characterize up to what extent each client seems to be benign or coherent with the rest. According to the results we achieved, FLDetector and FLTrust seem to be more stable than FLARE.

Acknowledgments. This study was funded by AEI Grant PID2020-119367RB-I00.

Disclosure of Interests. It is now necessary to declare any competing interests or to specifically state that the authors have no competing interests.

References

- Wang, Ning and Xiao, Yang and Chen, Yimin and Hu, Yang and Lou, Wenjing and Hou, Y. Thomas: FLARE: Defending Federated Learning against Model Poisoning Attacks via Latent Space Representations. In: *Proceedings of the 2022 ACM on* Asia Conference on Computer and Communications Security, pp. 946–958, (2022) https://doi.org/10.1145/3488932.3517395.
- Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong: FLTrust: Byzantine-robust Federated Learning via Trust Bootstrapping. 2022. arXiv preprint arXiv:2012.13995.
- Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov: How To Backdoor Federated Learning. 2019. arXiv preprint arXiv:1807.00459.
- Zhang, Zaixi and Cao, Xiaoyu and Jia, Jinyuan and Zhenqiang Gong, Neil: FLDetector: Defending Federated Learning Against Model Poisoning Attacks via Detecting Malicious Clients. KDD '22: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022.
- Zhang,Lefeng and Zhu, Tianqing and Zhang, Haibin and Xiong,Ping and Zhou, Wanlei: FedRecovery: Differentially Private Machine Unlearning for Federated Learning Frameworks. IEEE Transactions on Information Forensics and Security. pp 4732–4746. 2023
- Deng, Y. and M. Kamani, M and Mahdavi, M.:Adaptive Personalized Federated Learning. Preprint arXiv:2003.13461. 2020