BRIDGING THE OBSERVABILITY GAP: AUGMENTED REALITY POLICIES FOR HUMAN ROBOT COLLABORATION

BY

KISHAN DHANANJAY CHANDAN

B*E*, Mumbai University, 2017
M*S*, Binghamton University, 2019

DISSERTATION

Submitted in partial fulfillment of the requirements for
the degree of **Doctor of Philosophy** in Computer Science
in the Graduate School of
Binghamton University
State University of New York
2023

Accepted in partial fulfillment of the requirements for
the degree of **Doctor of Philosophy** in Computer Science
in the Graduate School of
Binghamton University
State University of New York
2023

May, 2023

Shiqi Zhang, Chair
Department of Computer Science, SUNY Binghamton

Lijun Yin, Member
Department of Computer Science, SUNY Binghamton

Adnan Siraj Rakin, Member
Department of Computer Science, SUNY Binghamton

Jivko Sinapov, Member
Department of Computer Science, Tufts University

Scott Craver, Outside Examiner
Department of Electrical and Computer Engineering, SUNY Binghamton

Abstract

In recent years, the use of Augmented Reality (AR) technologies are emerging as a promising solution to enhance human-robot collaboration by enabling a new medium for communication and interaction between human-robot teammates. When people and robots work in shared environments, it is vital that they communicate to collaborate with each other to avoid conflicts, leverage complementary capabilities, and facilitate the smooth accomplishment of tasks. Although the communication modalities from the literature have enabled robots and humans to exchange information, an observability gap exists due to the lack of effective sharing of the agents' observations. Robot and human agents have different sensory capabilities leading to a mismatch in their observations, and the agents need to communicate their observations toward effective human-robot collaboration. The real world is by-default 3D in nature, and AR provides a communication medium that enables an easy way to communicate spatial information by overlaying virtual objects over the real world leading to a comprehensive understanding of the surroundings for human-robot teams. This dissertation is centered around leveraging AR for bridging the observability gap between humans and robots.

In general, human-robot communication can be separated into two categories, 1, **proximate communication** - humans and robots are colocated, and, 2, **remote communication** - the human and the robot are not colocated, and are

separated spatially. In proximate communication in shared environments, humans and robots observe the environment directly, while in remote communication (shared embodiment) the human observes the environment via the robot. The main contributions of this dissertation are the algorithms and frameworks that enable human-robot teams to effectively collaborate with each other in both proximate and remote communication scenarios.

The frameworks specifically focus on leveraging AR for human-robot collaboration, and have enabled robots to integrate human feedback in planning, learn visualization policies for human-robot teams, and learned policies to guide human attention toward potential areas of interest. Our frameworks have been evaluated in the simulation and deployed on real robots with human participants working on tasks including collaborative delivery tasks (office and warehouse domains), and searching for target objects. The results indicate that the frameworks elevated the human-robot team efficiency, and provided a better user experience.

Dedicated to **Alpa Chandan** (Mom), **Dhananjay Chandan** (Dad),

and my dearest **grandparents**.

Acknowledgements

I am deeply grateful to Professor Shiqi Zhang for their invaluable guidance and unwavering support throughout my Ph.D. studies. Their constructive feedback and unwavering belief in my abilities have played a pivotal role in shaping the trajectory of my research. I will forever cherish the profound discussions, late-night paper edits, numerous demonstrations, and collaborative whiteboard sessions, as well as the abundant resources they provided to facilitate my research journey.

I extend my heartfelt gratitude to my exceptional committee members: Professor Yao Liu, Professor Lijun Yin, Professor Adnan Siraj Rakin, Professor Jivko Sinapov, and Professor Lei Yu. Your insightful feedback and brainstorming sessions have been instrumental in shaping the direction of my research. I would also like to express my sincere appreciation to Professor Scott Craver for graciously serving as the external examiner. Your expertise and support throughout the entire process are truly valued and deeply appreciated.

The computer science department has offered me an incredible opportunity, and the unwavering support from its members has allowed me to focus wholeheartedly on my research. Professor Weiyi Meng, I am immensely grateful for the time you dedicated to critical discussions and for providing invaluable feedback whenever I sought assistance.

To Professor Leslie Lander, I am grateful for the firsthand experience of your warm and welcoming nature. Regardless of the difficulties I faced, you were always there to ensure that I overcame them, standing by me every step of the way.

Professor Madhusudhan Govindaraju, I owe you an immeasurable debt of gratitude. You have been my pillar of strength when everything seemed impossible.

Since the beginning of my Ph.D. journey, you have consistently looked out for me, and I am forever grateful.

Jane, it's time for another happy dance. You epitomize the ideal mentor, exceeding all expectations. In our brief time working together, I have gained invaluable knowledge that I will carry with me for a lifetime. You possess the remarkable ability to bring out the best in everyone, and I am incredibly proud to call you my manager.

To my primary school teacher, Mrs. Sofi, my life story would have taken a different path if not for your unwavering confidence in me. I remember you with gratitude during every milestone in my life.

I will deeply miss the engaging discussions with my fellow lab mates. Saeid Amiri, our extensive conversations helped me start each day afresh, even when the light at the end of the tunnel seemed dim. Yohei Hayamizu, whether in the lab or at home, you were always there to lend an ear. You were an incredible friend and an amazing roommate. Yan Ding, I will forever cherish our whiteboard sessions and profound conversations. David Defazio, your unbiased advice has been invaluable. Whenever I needed assistance, you were there wholeheartedly. Since your arrival in the lab, Xiaohan Zhang, I have witnessed your growth, and I will always remember the coding sessions we shared at my place. Apart from my Ph.D. lab mates, I had the privilege of meeting exceptional minds such as Jack Albertson, Chelsea Zou, Eisuke Hirota, and Kevin Wallace. Your infectious energy and unwavering determination served as an inspiration to me. Thank you all for creating everlasting memories.

I have been fortunate to have dependable friends who have always been just a call away. Mit, Ankit, and Sagar thank you for standing by my side. I wish you could have been with me in the United States throughout this journey. Roger, our long drives and cooking sessions kept me motivated during this seemingly insurmountable journey. Ashish, you have been an integral part of my journey. Our profound phone conversations and philosophical discussions encouraged me

to persevere. Mayur and Kishan, you have always had my back, providing me with the courage to think outside the box.

I would like to take a moment to express my heartfelt gratitude to each of my amazing cousins for becoming an incredible support system in my life: Deepali, Pooja, Amit, Ashish, Parin, Bhavna, Naincy, Hitesh, Viral, Priyanka, Shubham, Deep, Niharika, and Jishnu. I cannot thank you enough for bringing your wonderful significant others into my life, and brightening it even more: Jikesh, Ruchi, Pooja, Divya, Nirav, Sunny, Reema, Jui, and Sourabh. At this moment, I would also like to share this milestone with Chetan, who I wish could be here to share the joy. I must not forget the amazing GenZ members of my family, including Angel, Vritee, Yashi, Hayaan, and our dear Barbie girl Dhyana. You inspire me to work harder, and your innocent smiles never fail to lift my spirits.

Finally, my mom and dad have been my pillars of strength, and their sacrifices have paved the way for my achievements. I cannot thank them enough, as I am simply a reflection of their unwavering dedication. Khushboo, thank you for supporting my decision to pursue my dreams in a distant country while taking care of mom and dad. Without your support, enrolling in a Ph.D. program would have been inconceivable. Vidisha, my knight in shining armor, thank you for being the beacon of hope when everything seemed to crumble. As my partner in crime and my biggest critic, I would not be half as capable without your help in overcoming challenges. My extended family has been my backbone, providing immense support at every turn. Their unwavering belief in me has driven me to improve each day. Lastly, to Cosmo, my dog, you never failed to bring a smile to my face in the toughest of times. No matter how difficult the day, you always ensured my well-being. Each of you has contributed to making me a better person.

I would like to acknowledge the countless other individuals who have contributed to this milestone. Without you amazing people, this achievement would not have been possible. I extend my sincere gratitude to every one of you for being by my side. I consider myself incredibly lucky to have you all in my life.

Contents

List of Tables

List of Figures

1 Introduction

Robots are increasingly ubiquitous in everyday environments including offices, warehouses, hotels, as well as homes. The advancements in robot autonomy have enabled the robots to execute long-horizon tasks without the need for human supervision. While the robots can accomplish a wide variety of tasks autonomously, there are still scenarios where the robots need human help. For instance, the robots might lack the physical capabilities to accomplish a task like opening a door, or the robots might lack the capabilities to reason about how to accomplish the task in case of unforeseen issues at task execution time. Human-robot collaboration is becoming increasingly important as robots become more prevalent in everyday environments.

In the context of human-robot collaboration as a team, the actions of every agent, both robots and humans, to achieve their goals, have a direct effect on collaboration efficiency. Robots and humans perceive the environment in different ways, and have different levels of situational awareness, as well as, a different understanding of the environment around them. It is crucial that the agents communicate their observabilities, to ensure that each agent's observability is considered for joint decision-making of the team. Towards achieving efficient human-robot collaboration, the agents also need to effectively share task-related information, such as their intermediate tasks, towards achieving their shared end goals.

On the one hand, humans are inclined towards using natural language and gestures to communicate, while the robots better understand digital forms like text-based commands, which results in a communication gap. To alleviate this, researchers have employed different communication modalities such as natural language, gestures, and haptics to enable humans and robots to interact with each other, and share information [1]. Augmented Reality (AR) allows the user to see the real world, with virtual objects superimposed upon or composited with the real world supplementing the reality, where the virtual and real objects coexist in the same space [2]. Over the years, AR has been employed in different application scenarios like education, medicinal surgeries, industrial training and safety, rehabilitation, warehouses, etc [3].

Both humans and robots observe different aspects of the world, and, the advancements in human-robot communication facilitate the human and robot agents to share their observations, but there still exists a significant gap in their observability. For example, in the context of telepresence systems, the robot has broader observability over the environment, whereas the human might have partial observations compared to the robot. In this case, there exists an observability gap, where even though both humans and robots share the same embodiment, their perception of the environment and their situational awareness may be different. Another example of an observability gap can be during human-robot collaborative delivery tasks, where the robot and human agents might not be able to directly observe each other's task status due to occlusions (e.g. walls), making it difficult to effectively collaborate.

The observability gap is the difference in capabilities of observing

a world between agents (humans and robots), that potentially lead to different individually-believed-optimal plans of achieving the same goal. Bridging the observability gap helps improve the overall observability over the world of the whole human-robot team.

By bridging the observability gap, we can bring the understanding of the human and robot agents a step closer to full observability. Toward bridging the observability gap, we utilize AR for enabling human-robot communication and increase the efficiency of human-robot collaboration. I present the different frameworks and algorithms designed in the following subsection and describe the research theme.

1.1 Research Theme

Although AR has a rich history of enabling human and robot agents to communicate with each other, there still exist limitations. For instance, the existing AR-based systems that facilitate human-robot communication have leveraged AR visualizations to provide the human agent with the information necessary for task completion. While these systems enable the human agent to observe the robot's states, these systems assume that the human agent can proactively adapt their plan by leveraging the information from AR visualizations. This shows that there exists an observability gap from the human agent to the robots, because the human agent's information/feedback can be useful for the robot agents to accomplish tasks. This research falls under the proximate communication category where I can focus on enabling bi-directional communication in human-robot teams working on collaborative delivery tasks in an office environment.

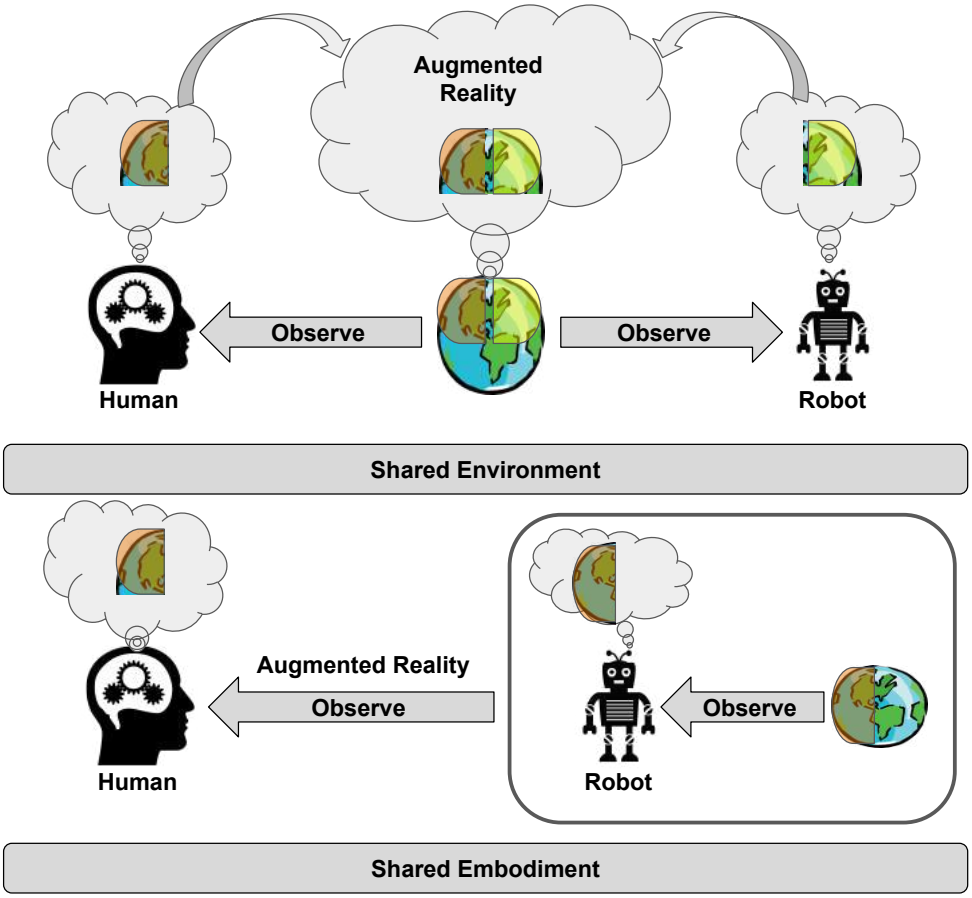


Figure 1.1: An overview figure highlighting my contributions in different dimensions. The dissertation focuses on bridging the observability gap for both proximate (shared environment) and remote communication (shared embodiment).

Additionally, all the existing visualization policies for AR interfaces from the literature are handcrafted and static, which does not consider the state of the human-robot system to effectively display visualizations. Apart from enabling humans to share their feedback with the robots, I focus on learning visualization policies that dynamically adapt based on the state of the human-robot system, specifically in the warehouse domain, where human and robots work on collaborative delivery tasks. Again this focuses on the proximate communication category, where the human and robot agents work in a shared environment, and AR mediates the communication between the agents as seen in Figure 1.1 (top).

When robots and humans work together in a telepresence setting, the situational awareness of the human operator depends on the information observed via the robot. I focus on leveraging AR visualizations to elevate the situation awareness of the human operator in remote communication, where the human and robot agent share embodiment, and the human observes the remote world via the robot (Figure 1.1 bottom).

It is to be noted that in all of our contributions, we assume that the end goal of both the robot and human agents is shared, while the intermediate goals can differ. Moreover, another assumption in our contributions is that, the human teammate is cooperative and utilizes the AR visualizations towards efficient task completion. This dissertation aims to bridge the observability gap in different dimensions by improving the observability of the whole human-robot system aiming towards full observability, but none of our contributions claim full observability of the world. Next, I describe the individual contributions of this dissertation, and the respective frameworks that I designed to bridge the observability gap.

1. Employing AR for bidirectional communication: Researchers have enabled bi-directional communication in human-robot teams via AR interfaces. Although both robots and human teammates can communicate their status with the team, the information from the human teammate is not utilized by the robots creating an observability gap. To address the observability gap, we designed a framework, called ARROCH, that bridges the observability gap between the human and robot agents by providing the robots' state information to the human agent, and additionally creating a bidirectional communication channel to enable the human to provide task-related information in terms of feedback.

ARROCH bridges the observability gap by enabling the human-robot teams to observe the other agents' plans. The observability of the plans enables the agents to infer the other agents' future task information in AR. Our ARROCH framework enables the human agent to visualize the current state, and the next set of actions of a team of robots in AR, while also allowing the human to provide feedback on the robots' plans. More importantly, as an advancement from the existing systems from literature, ARROCH allows the robots at runtime to integrate human feedback towards efficient tasks accomplishments. Additional details of ARROCH are provided in Chapter 4.

2. Learning AR policies from demonstrations: In my first contribution, I enabled the human agent to track the state and plan of a team of robots using AR visualizations. While the visualizations prove helpful for the human to efficiently collaborate with the robots, a new challenge arises when the number of robots in the team increases. With the increase in the number of robot teammates, the AR visualizations can become overwhelming for the human agent to process, and effectively utilize for task completion. When too little information is visualized, human users find the AR interface not useful; when too much information is visualized, they find it difficult to process the visualized information creating an observability gap. Moreover, current AR-based visualization policies are designed manually, which requires a lot of human effort and domain knowledge. To address this challenge, we designed a framework, called, VARIL, that learns a visualization policy using imitation learning that decides what to visualize, when, and how from human expert demonstrations. VARIL utilizes the human-robot team's state information to learn a visualization policy. We provide more details of VARIL

in Chapter 5.

3. Learning AR policies from reinforcements: In the VARIL framework, we have employed imitation learning to enable the visualization agent to learn a visualization policy. Our framework naturally inherits the demerits of imitation learning, where the performance is bound by the performance of the human expert providing demonstrations. Additionally, collecting a dataset with human demonstrations is costly. Reinforcement learning has enabled robots to learn from interaction with the environment. To tackle the shortcomings of VARIL, I leverage multi-agent reinforcement learning to learn visualization policies of AR in the human-multi-robot setting, and design a framework called VARMAL. We utilize the value-based method (QMIX) to train the visualization policies in a centralized end-to-end fashion, while deploying them in a decentralized manner. We provide more details of VARMAL in Chapter 7.

4. Benchmarking AR agent for large-scale experiments: The complexity of a human-robot system exponentially increases as we move from a human-single-robot system to a human-multi-robot system. Adding to this complexity, the AR agent elevates this problem because the AR agent acts as a communication medium between the human-multi-robot teams. Such a complex system is very challenging to systematically evaluate due to the large number of interdependent agents working towards their shared goal or even their individual goals. Moreover, the wide existing literature on simulation platforms does not provide the functionality of AR. To address this shortcoming, I design a simulator that simulates the human-multi-robot teams, and their interactions mediated by AR. The simulator provides a way to model the behavior of the virtual human, and

change the manner in which the virtual human reacts to the visualizations of AR. Most importantly, the simulator provides different kinds of visualization policies, including the adapted implementations of some systems from the literature. We present this simulator as a benchmark platform to help the growing community of AR researchers working in the robotics domain to jump-start their research and deploy their AR interfaces in simulation without the added burden of working with multi-robot systems or additional AR hardware.

5. Learning to guide human attention: In the case of remote communication, the human shares the same embodiment as the robot, where the human is located in a remote environment. It is vital that both the human and the robot agents share high levels of situational awareness while sharing the same embodiment in shared autonomy tasks. Advancements in sensing technologies have enabled robots to observe the entire 360-degree field of view of the surroundings. In the context of telepresence, such an all-degree view can be helpful for humans to gain situational awareness closer to the robots. In general, the human field of view spans only 120 degrees, making it difficult for humans to perceive all the available information. The difference in the robot's and human's level of situational awareness, and the lack of human's ability to perceive the available information to the level of robots' observabilities leads to an observability gap. To address the observability gap created by the different levels of observabilities in terms of their field-of-view, I designed a framework, called GHAL360, that tracks the all-degree view of the remote environment, and accurately guides the human attention using visualizations toward the area of interest. In GHAL360, the actions of both the robot and the human agent are shared due to the same embodiment. We learn

the visualizations of GHAL360 using reinforcements to effectively guide human attention, when the area of interest is out of the field of view of the human agent. Further details on "GHAL360" are provided in Chapter 6.

1.2 Dissertation Organization

In Chapter 2, I provide the background information that is the basis of different components used in the frameworks that are part of the core contributions of this dissertation. Later, in Chapter 4, we provide a thorough literature review of the different communication modalities, and the parallel works that align with the theme of leveraging AR for human-robot communication. In the following Chapters 4, 5, 7, 6, and 8, we delineate our individual contributions. Finally, in Chapter 9, I summarize the contributions and then delineate the future work in Chapter 10.

2 Background

This section provides background information about the dissertation. It contains information about Answer Set Programming (ASP), and different machine learning methodologies including MDPs, Supervised Learning, and Reinforcement Learning.

2.1 Answer Set Programming

Answer Set Programming (ASP) is a popular non-monotonic logic programming paradigm that utilizes declarative language for knowledge representation and reasoning (KRR) [4, 5], and has been applied to a variety of planning problems [6], [7], [8], including robotics [9]. ASP is particularly useful for robot task planning in domains that include a large number of objects [31]. An ASP program consists of a collection of statements, that provides the domain description, including the objects and their relations. Defining a domain in ASP syntax requires the core ASP elements of atoms, literals, and rules. An atom is an elementary proposition, such as l, while a literal is an atom with its negation, such as l and -l. Literal can also be the default negation of an atom represented by not. ASP provides the ability to perform default reasoning using concepts such as default negation and epistemic disjunction, e.g., unlike "¬ a", "not a" only implies that "a is not believed to be true" and does not imply that "a is believed to be false". A rule

consists of a head and a body. The head is true when the body is true. A rule is expressed as: $l_0, \cdots, l_k$:- $l_{k+1}, \cdots, l_m$, not $l_{m+1}, \cdots,$ not l_n where l_i is a literal. A rule without the head is a constraint, and a rule without the body is a fact. Solving an ASP program results in an answer set, which contains a set of all the possible solutions. An answer set is a set of ground literals that represent the beliefs of an agent associated with the program.

2.2 Machine Learning Methodologies

Machine learning (ML) is used to teach machines how to handle data more efficiently. Machine Learning consists of different algorithms to solve data problems. Some of the commonly used algorithms in ML are Supervised Learning, Unsupervised Learning, Semi-supervised Learning, Reinforcement Learning, and Ensemble Learning. In this section, we will briefly describe Supervised Learning, and Reinforcement Learning that are leveraged in this dissertation.

2.2.1 Markov Decision Processes

Markov Decision Processes (MDPs) [10] provide a mathematical framework for modeling decision-making in situations where outcomes are partly random and partly under the control of a decision-maker. The dynamics of the environment can be modeled in the form of an MDP to equip the agent with decision-making capabilities. Unlike classical planning paradigms, where given a start and an end goal, the planner provides a plan for the agent, solving MDP results in a policy, that contains a mapping from the current state to action. MDP is represented by a tuple $\langle S, A, T, R \rangle$ where

- S is a set of states

- A is a set of actions.
- $T(s, a, s') = P(s'|s, a)$, where in each state $s \in S$, the agent takes an action $a \in A$ and reaches a new state s', determined from the probability distribution $P(s'|s, a)$. If $P(s'|s, a) = \{0, 1\}$ the system is said to be *deterministic*, otherwise it is *stochastic*.
- $R(s, a)$ is the immediate reward, the agent receives on reaching the new state s'.

2.2.2 Partially Observable Markov Decision Processes

A generalized form of an MDP, where the assumption is that the world is partially observable is termed a Partially Observable Markov Decision Processes (POMDP) [11].

A POMDP can be described as a tuple $\langle S, A, T, R, \Omega, O \rangle$, where,

- S, A, T, and R describe a MDP;
- Ω is a finite set of observations the agent can experience of its world;
- O: S x $A \rightarrow \Pi(\Omega)$ is the observation function, which gives, for each action and resulting state, a probability distribution over possible observations, where $0(s^{'}, a, o)$ is the probability of making observation o given that the agent took action a and landed in state $s^{'}$.

A POMDP is an MDP in which the agent is unable to observe the current state, and it relies on observation based on the action and resulting state.

2.2.3 Supervised Learning

Supervised Learning (SL) is a machine learning paradigm that enables the agent to learn a policy using a labeled dataset [12]. The labeled dataset contains

a large set of features and their relative labels, which are used in the training phase to learn a mapping from a particular state to an action.

Imitation Learning: Imitation Learning (IL) is a type of SL, where the agent learns the policy by mimicking the demonstrations provided by an expert [13]. Typically, IL problems are represented in the form of an MDP, where the agent observes the current state, and the action suggested by the expert, and tries to learn a policy as close to the expert as possible.

2.2.4 Reinforcement Learning

In Reinforcement Learning (RL), an agent interacts with an environment E by observing the current state and executing an action in discrete time steps [14]. Once an action is executed the state changes (the change can be either deterministic or stochastic), the agent receives a reward and the process repeats until the agent reaches the terminal state.

We can formulate this as an MDP. The optimal value function $Q^*(s, a)$ provides maximal values in all states and is determined by solving the Bellman equation:

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q^*(s',a'), \tag{2.1}$$

where $0 < \gamma < 1$ is the discount factor. The optimal policy π is then as follows:

$$\pi(s) = argmax_a Q^*(s,a). \tag{2.2}$$

RL methods are categorized into two main classes: model-based and model-free methods. One of the widely used model-free approaches is where the Q-values

are updated for each state and action pair whenever they are visited. Due to this, the agent needs to visit each state and action pair infinitely often for the value function to converge.

One algorithm for solving this problem is Q-learning [15], which minimizes the Bellman error of the Q function where optimal policy π^* is iteratively approximated by updating the estimated Q-values:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(R + \gamma \max_{a'} Q(s',a') \tag{2.3}$$

where α is the learning rate,and $1 < \alpha > 0$.

3 Related Work

In this chapter, I provide a literature review of the different related existing research including various human-robot communication modalities, applications of learning in robotics, telepresence systems, and the rich literature on embodied AI agent platforms.

3.1 Human-Robot Communication Modalities

When humans and robots work in proximity, there is a need for communication to ensure the safety and efficiency of HRC. In this section, we first describe a few communication modalities used in current human-robot systems (non-AR and AR)

3.1.1 Traditional Communication Modalities

Humans and robots prefer different communication modalities. While humans employ natural language and gestures, the information in digital forms, such as text-based commands, is more friendly to robots. Researchers have developed algorithms to bridge the human-robot communication gap using natural language [16, 17, 18, 19] and vision [20, 21, 22]. Despite those successes, AR has its unique advantage of providing a communication medium with potentially less ambiguity and higher bandwidth [2]. AR helps in elevating coordination through communicating spatial information, e.g., through which door a robot is coming into a room and

how (i.e., the planned trajectory), when people and robots share a physical environment. Researchers are increasingly paying attention to AR-based human-robot systems [23].

Projection-based Communication: One way of delivering spatial information related to the local environment, referred to as "projection-based AR," is through projecting the robot's state and motion intention to the humans using visual cues [24, 25, 26]. For instance, researchers used an LED projector attached to the robot to show its planned motion trajectories, allowing the human partner to respond to the robot's plan to avoid possible collisions [27]. While projection-based AR systems facilitate human-robot communication about spatial information, they have the requirement that the human must be in close proximity to the robot.

Researchers have used 2D interfaces in the past where the communication usually takes place using graphical user interfaces [28, 29, 30]. Also, natural language was used by a number of researchers for human-robot communication [19, 17]. Motion-based signaling of intention like gestures, including hand and facial has also been used for communication in human-robot teams [31, 32, 33]. Moreover, researchers studied how non-verbal social cues can improve task performance in human-robot teams [34]. Haptics, another communication modality, is largely based on physical interaction [35, 36, 37]. Even though all these modalities have their unique benefits and provide a medium for human-robot communication, the real world is 3D by nature, and these modalities are not strong in conveying spatial information, which can be very useful for HRC in shared spaces.

3.1.2 Augmented Reality Interfaces

AR has the capability of overlaying spatial information onto the real world and helping a human counterpart visualize the robots' internal state and intended plans for effective HRC. Early applications of AR in robotics include a system called, ARGOS, which enabled the human to track the robot's status and also teleoperate it [38]. Over the years, AR has been employed in different application scenarios like education, medicinal surgeries, industrial training and safety, rehabilitation, warehouses, etc [3]. Researchers have leveraged AR to enhance the students' learning experience [39, 40]. In another line of research, Cheli et al., build a framework that investigates how AR can catalyze students' ability to understand robot behavior [41]. Similarly, Zhang et al. [42] used AR paired with a Lego robot to expose pre-college level students to the Q-learning reinforcement learning algorithm and human-in-the-loop training.

The bounds of AR in education have expanded to more complex tasks like industry training and guidance, where the employees can leverage AR to learn to work with robots faster, and with an increase in safety [43, 44]. On the one hand, Lai et al. designed a system to enhance worker's performance [45] using AR instructions, while on the other hand, AR was used to create a monitoring system in assembly lines [46]. Moreover, researchers have utilized AR-based systems for elevating human-robot collaboration in industry [47, 48, 49], and logistics [50].

Active investigations of the use cases of AR for robotic surgeries[51] are being conducted in the field of medicine. AR has proved useful in training as well as in assisting during surgeries [52, 53, 54, 55]. Researchers have designed AR-based systems that can help in surgeries by leveraging hyper-accuracy three-dimensional

reconstruction technologies [56]. The usefulness of AR in the field of medicine is evident from the evolving literature [57, 58, 59, 60, 61, 62]. The increase in the feasibility of using AR in medicine is the result of the new generation AR devices that have significantly better accuracy and speed as opposed to their predecessor.

With the advancement of AR devices, AR-based human-robot interaction (HRI) research has boomed. Researchers have enabled a human operator to interactively plan, and optimize robot trajectories [63, 64]. More recently, researchers have developed frameworks that enable human operators to visualize the motion-level intentions of robots using AR [65, 66, 67, 68, 69]. AR-based systems have also been used to improve the teleoperation of collocated robots [66]. Recently, researchers have studied how AR-based visualizations can help the shared control of robotic systems [68]. Muhammad et al. created a system that enables the human user to visualize the robot's sensory information, and planned trajectories, while allowing the robot to ask questions through an AR interface [70].

More recently, researchers have designed different visual guidance in AR, where the humans can visualize the recommended actions from the robot (prescriptive guidance), and also visualize the state space information (descriptive guidance) to communicate why the robot recommends a particular action to the human [71]. One common limitation of those systems is their unidirectional communication, i.e., their methods only convey the robot's intentions to the human but do not support the communication the other way around. Due to this setting, the human teammate is expected to adapt by visualizing the robot's status.

Researchers have enabled bi-directional communication in AR and have studied how AR-based visualizations can help the shared control of robotic systems [68].

Researchers have also designed a system that helps a human user to visualize the robot's sensory information, and planned trajectories, while allowing the robot to ask questions through an AR interface [70]. In another line of research, Zhu et al. augment the algorithms that control the robots to help understand the relationship between the robot's state and the environment [72]. Researchers have also developed a prototype system with a shared AR workspace that enables a shared perception [73].

Design of AR Interfaces: The research community working on AR for HRI lacks a set of shared terminology and framework for characterizing aspects of AR interfaces. A recent survey paper focuses on creating a common set of terms and concepts with a novel taxonomic framework for different types of VAM-HRI interfaces [74]. The VAM-based visualizations are termed as virtual design elements (VDEs) and are categorized to aid interface design. More recently, researchers have created a set of salient VDEs for enhancing the human perception of the robot's capabilities [75]. Researchers have developed a dynamic AR path visualizer that projects the robot's motion intent at varying lengths depending on the the complexity of the upcoming path [76].

One of the main research areas that is gaining a lot of attention is view management for AR interfaces. There are known problems like visual cluttering due to the augmentation of many virtual objects, visibility issues due to overlapping of 3D objects, label and annotation placements over 3D objects, etc. The arrangement of objects in an AR interface can largely affect the understanding of the augmented scene. Visual clutter is a problem associated with the arrangement of objects, and can lead to degradation in task performance [77]. To overcome the

problem of visual clutter, some researchers have opted for filtering techniques [78]. Other approaches include spatial rearrangement of labels to avoid overlapping and to reduce visual clutter [79, 80, 81].

3.2 Learning in Robotics

Learning has been applied to different robotics domains to teach the robots skills that were not possible before. In the following subsections, I will describe the different applications of learning, both Imitation Learning and Reinforcement Learning in robotics.

Imitation Learning in Robotics: The integration of Imitation Learning (IL) with robotics has enabled the robots to learn a variety of tasks by observing expert demonstrations [13]. One of the early applications of IL was to teach helicopters to fly very challenging maneuvers through providing expert demonstrations [82]. In another line of work, IL was used to enable a vehicle to mimic human driving behaviors in outdoor terrains [83, 84]. Later on, researchers used IL to learn from expert demonstrations in a simulated car driving domain [85]. A comprehensive study has been carried out to showcase the use of IL in robotics [86]. Very recently, researchers have used AR interfaces for constrained Learning from Demonstration (LfD) to allow the agents to maintain, update, and adapt learned skills [87].

Reinforcement Learning in Robotics: The integration of Reinforcement Learning (RL) with robotics has enabled the robots to learn a variety of tasks by interaction with the environment [14, 88]. Reinforcement learning enables a robot to autonomously discover an optimal behavior through trial-and-error interactions with its environment. Learning to grasp a ball in RoboCup [89], robotic arm ob-

ject grasping [90], and pouring liquid [91], are among the many tasks learned by robots using RL. RL has been extended to learn optimal collaborative behaviors in HRC. Researchers proposed an RL-based framework for human-robot collaborative manipulation tasks [92]. The robot learned to reach and hold the table, and keep the table horizontal during lifting it up with humans in a reasonable amount of time. In another line of research, an RL-based variable admittance controller is proposed for human-robot co-manipulation task [93]. Some researchers also designed a multi-sensing framework for personalized HRC and Assistive Training using RL [94].

Learning in AR: AR has also been leveraged for learning in robotics systems. In one line of research, researchers designed a system for helping users give improved trajectory demonstrations for a robotic arm using AR visualizations [95]. To improve this system, AR-based counterfactual explanations from explainable AI literature was used to generate 'what-if ' visualizations, allowing human operators to reason about the validity of the model learned by the system [96]. In another research, a system was designed for ensuring reproducibility when transferring a demonstrated motion to a system by guiding the user during the demonstration, preventing them from demonstrating non-reproducible motions [97]. An AR-based tool was designed to improve sim2real reinforcement learning for robots by allowing the user to visualize the robot's training data in the context of the environment to provide insights into ways to improve the robot's training process [98]. In another line, deep reinforcement and imitation learning were combined in an AR-based system to improve adaptation to changing scenarios for robotic bin picking[99].

3.3 Telepresence Systems

There is a rich literature on the research of mobile telepresence robots (MTRs). Early systems include a telepresence system using a miniature car model [100], and a telepresence robot for enabling remote mentoring, called Telementoring [101]. Those systems used monocular cameras that produce a narrow field of view over remote environments. MTR systems were developed to use pan-tilt cameras to perceive the remote environment [102, 28, 103], where the human operator needs to control both the pan-tilt camera and the robot platform. To relieve the operator from the manual control of the camera, researchers leveraged head motion to automatically control the motion of pan-tilt cameras using head-mounted displays [104, 105]. While such systems eliminated the need of controlling a pan-tilt camera's pose, they still suffered from the limited visual field.

To increase a robot's field of view, researchers have equipped mobile robots with multiple cameras [106, 107], and wide-angle cameras [108, 109]. Recently, researchers have developed MTRs with 360° vision systems [110, 111]. Examples include the MTRs with 360° vision for redirected walking [112], and virtual tours [113]. It should be noted that transmitting 360° visual data brings computational burden and bandwidth requirement to the robot [114, 115].

A rich literature exists regarding active vision within the computer vision and robotics communities [116, 117, 118], where active vision methods enable an agent to plan actions to acquire data from different viewpoints. For instance, recent research has showcased that an end-to-end learning approach enables an agent to learn motion policies for active visual recognition [119]. Researchers have also designed a system that automatically controls a virtual camera to choose and

display the portions of video inside the 360° video [120, 121].

3.4 Robotics Simulation Platforms

In this subsection, I will briefly provide an overview of the existing open-source simulators for HRI research based on their functionalities. I focus on the rich literature on embodied AI platforms, where the focus is to enable more natural and intuitive interactions between humans and robots.

Advancement in robotics has led to the development of several robotic simulators. One of the earliest and most widely used simulators in the robotics community is Gazebo [122], which supports a lot of robotic platforms and has tight coupling with Robot Operating System (ROS) [123], again a middleware well-known in robotics. The development of these simulators helps to eliminate the difficulty of building robot agents and 3D environment models. Moreover, it saves the cost of expensive robotic hardware and also saves the maintenance cost. Finally, many of these simulators allow increasing the sim time to speed up the robots, enabling them to run a large set of experiments as compared to the real robots.

ThreeDWorld is another simulator that enables the simulation of high-fidelity sensory data and physical interactions between mobile agents and objects in rich 3D environments, and also provides a medium for data collection in VR [124]. Very recently, researchers have designed a user-centric embodied AI platform, called Alexa Arena, that focuses on efficient data collection, and provides a web-based user interface allowing real-time user communications and interactions via natural language, and/or keyboard and mouse. The emergence of new realistic 3D datasets

of real environments like Matterport3D [125] and SUNCG [126] paved the way for several photorealistic simulators like HoME [127], MINOS [128], Gibson [129].

Yan et al. designed a multi-room 3D house simulator while enabling the agent to interact with objects and move between rooms [130]. VirtualHome is another 3D simulator that focuses on creating large video activity datasets for household activities [131]. Another widely used embodied agent platform, called, AI2THOR is a near-photorealistic simulator for indoor environments with interactive objects [132]. Furthermore, a simulation environment, called SAPIEN, was developed for robotic vision and interaction tasks [133]. For enabling agents to learn in realistic environments, VRKitchen was developed that facilitates reinforcement learning, learning from demonstrations for complex tasks in the kitchen environments [134]. A sophisticated 3D simulator, called Habitat 2.0 [135, 136], can create digital twins of the real world which enables efficient experimentation involving embodied AI agents rearranging richly interactive 3D environments. A cloud-based VR platform, named SIGVerse, was developed to reduce the cost of developing real robots and human-robot interaction experiments in the real world [137].

Very recently, there have been several simulators incorporating AR into the simulation as a communication medium. A VR simulation system was developed for service robots in an office environment for immersive interaction experience between virtual robots and humans using interfaces based on AR, speech recognition as well as gaze and body tracking technologies [138]. Haven [139] is yet another simulator that focuses on HRI mediated by AR, where the human can visualize the motion plan, as well as the robot's internal status like remaining battery power. Very recently, researchers have studied different VDEs for AR in a

simulated environment to help the robot express its abilities in navigation, natural language processing, and audio localization, to help the human better understand the robot's capabilities.

Finally, I present some additional dimensions to compare the existing literature on the embodied AI agent platforms in Table 3.1.

Table 3.1: A summary of the existing simulation platforms for embodied AI agents comparing the support for, 1)robotic middleware (ROS), 2, human-robot interaction, 3, multi-robot system, 4, task planning capabilities, 5, deployment on AR/VR devices.

	Robotic middleware	HRI	Multiple robots	Task planning	AR/VR devices
HoME [127]	✗	✗	✗	✗	✗
VirtualHome [131]	✗	✗	✗	✗	✗
AI2THOR [132]	✗	✓	✓	✓	✓
MINOS [128]	✗	✗	✗	✗	✗
SAPIEN [133]	✓	✗	✗	✗	✗
SIGVERSE [137]	✓	✓	✗	✗	✗
iGibson [129]	✓	✗	✓	✗	✗
VRKitchen [134]	✗	✓	✗	✓	✓
Habitat 2.0 [135]	✗	✗	✗	✗	✗
CHALET [130]	✗	✗	✗	✗	✗
ThreeDWorld [124]	✗	✓	✓	✗	✓
Alexa Arena [140]	✗	✓	✗	✓	✗
Haven [139]	✗	✓	✗	✗	✓
Reimagining RViz [75]	✗	✓	✗	✗	✓
MAARS (ours)	✓	✓	✓	✓	✓

4 Employing AR for Bidirectional Communication

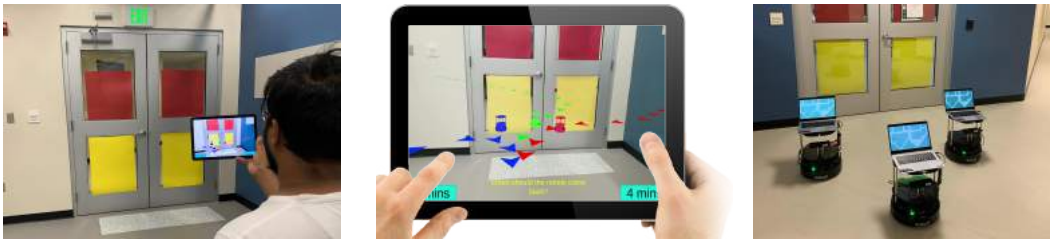
In this chapter, I discuss my contribution to augmented reality-mediated bidirectional communication for human-robot collaboration. The human teammate can leverage the AR interface to see through obstacles to observe the robots' current states and intentions, and provide feedback, while the robots' behaviors are then adjusted toward human-multi-robot teamwork. This contribution falls under the proximate communication type, where human and robots are collocated.

4.1 Introduction

Robots are increasingly present in everyday environments, such as warehouses, hotels, and airports, but human-robot collaboration (**HRC**) is still a challenging problem. As a consequence, for instance, the work zones for robots and people in warehouses are separated [141]; and hotel delivery robots barely communicate with people until the moment of delivery [142]. When people and robots work in shared environments, it is vital that they communicate to collaborate with each other to avoid conflicts, leverage complementary capabilities, and facilitate the smooth accomplishment of tasks. In this chapter, we aim to enable people and robot teams to efficiently and accurately communicate their current states and intentions toward collaborative behaviors.

AR technologies focus on visually overlaying information in an augmented layer over the real environment to make the objects interactive [143]. AR has been applied to human-robot systems, where people can visualize the state of the robot in a visually enhanced form [144]. Most existing research on AR-based human-robot collaboration focuses on the visualization of robot status, and how people leverage the augmented information to adapt to robot behaviors, resulting in unidirectional communication [65, 24]. Another observation is that many of those methods from the literature were limited to human-single-robot, in-proximity scenarios [49, 145]. In this chapter, we focus on **human-multi-robot**, **beyond-proximity** settings, where the robots need to collaborate with each other and with people at the same time, in indoor multi-room environments. Our developed algorithm (and system) is called **ARROCH**, short for *AR for robots collaborating with a human.*

ARROCH supports **bidirectional** human-robot communication. On the one hand, ARROCH enables human to visualize the robots' current states, e.g., their current locations, as well as their intentions (planned actions), e.g., to enter a room. For instance, a human might see through an opaque door via AR to "X-ray" a mobile robot waiting outside along with its planned motion trajectory. On the other hand, leveraging a novel AR interface, ARROCH allows the human to give feedback to the robots' planned actions, say to temporarily forbid the robots from entering an area. Accordingly, the robots can incorporate such human feedback for replanning, avoiding conflicts, and constructing synergies at runtime. ARROCH supports beyond-proximity communication by visually augmenting the robots' states and intentions, which is particularly useful in indoor multi-room environments.



(a) Visualization of robots' states and intentions

(b) AR interface of ARROCH

(c) Robots waiting to enter a room

Figure 4.1: The AR interface of ARROCH enables the human to visualize the robots' current states (their current locations in this example) using 3D robot avatars, and their intentions (entering the room) using trajectory markers. ARROCH also enables human to give feedback to robot plans. In this example, the human can use the interactive buttons in the bottom corners to indicate he could not help open the door in a particular time frame, say "4 minutes."

ARROCH (algorithm and system), as the **first contribution** of this framework, has been evaluated in simulation and using real robots. The robots help people move objects, and the human helps the robots open doors (the robots do not have an arm, and cannot open the doors). Results from the real-world experiment suggest that ARROCH significantly improves the efficiency of human-robot collaboration, compared with a standard non-AR baseline. The simulation platform is constructed using Unity [146] for simulating the AR interface and human behaviors, and using Gazebo [122] for simulating robot behaviors. To the best of our knowledge, this is the first open-source simulation platform for simulating AR-based human-multi-robot behaviors, which is the **second contribution** of this research. In the simulation, ARROCH significantly improved human-robot teamwork efficiency in comparison to baseline methods with different communication mechanisms.

4.2 ARROCH Algorithm

In this section, we describe ARROCH (*AR for robots collaborating with a human*), an algorithm and system that utilizes AR technologies to enable bidirectional, multi-turn, beyond-proximity communication to enable collaborative behaviors within human-multi-robot teams. Figure 4.1 shows our novel AR interface, which enables the human to visualize the robots' current states, and their intentions (planned actions), while at the same time supporting the human to give feedback to the robots. The robots can then use the feedback to adjust their plans as necessary. Next, we describe the ARROCH algorithm, and then its system implementation.

4.2.1 The ARROCH Algorithm

The input of Algorithm 1 includes, $\boldsymbol{\mathcal{S}}$, a set of initial states of N robots, where $s_i \in \boldsymbol{\mathcal{S}}$ is the initial state of the i^{th} robot; $\boldsymbol{\mathcal{G}}$, a set of goal states, where $g_i \in \boldsymbol{\mathcal{G}}$ is a goal state of the i^{th} robot.[1] A multi-robot task planner, $\mathcal{P}^t$, and a motion planner, $\mathcal{P}^m$ are also provided as input.

ARROCH starts by initializing: an empty list of robot configurations (in *configuration space*, or C-Space), $\boldsymbol{\omega}$, where each configuration is in the simple form of $\langle x, y, \theta \rangle$ for our mobile robots, an empty list, $\boldsymbol{\mathcal{I}}$, to store the intended trajectories of the N robots in C-Space ($N = |\boldsymbol{\mathcal{S}}|$), and an empty list to store the activated constraints, $\boldsymbol{\mathcal{C}}$.

In Line 2, ARROCH generates a set of N plans, $\boldsymbol{P}$, using multi-robot task **Planner** $\mathcal{P}^t$. Entering the while-loop (Lines 3-17), ARROCH runs until $\sum_{p \in \boldsymbol{P}} |p| >$

[1] Strictly speaking, tasks are defined as goal conditions [147]. Here we directly take goal states as input for simplification.

Algorithm 1 ARROCH

Input: $\boldsymbol{\mathcal{S}}$, $\boldsymbol{\mathcal{G}}$, $\mathcal{P}^t, \mathcal{P}^m$

1: Initialize empty lists: $\boldsymbol{\omega} \leftarrow \emptyset$; $\boldsymbol{\mathcal{I}} \leftarrow \emptyset$; $\boldsymbol{\mathcal{C}} \leftarrow \emptyset$
2: $\boldsymbol{P} = \mathcal{P}^t(\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{G}}, \boldsymbol{\mathcal{C}})$, where $p \in \boldsymbol{P}$ is a task plan (an action sequence) for one robot, and $|\boldsymbol{P}| = N$ ▷ Task planner $\mathcal{P}^t$ in Section 4.2.3
3: **while** $\sum_{p \in \boldsymbol{P}} |p| > 0$ **do**
4: **for** $i \in [0, 1, \cdots, N\text{-}1]$ **do**
5: **if** p_i is **not** empty **then**
6: $a_i \leftarrow p_i$.front() ▷ Obtain current robot action
7: Obtain i^{th} robot's current configuration: $\boldsymbol{\omega}_i \leftarrow \theta(i)$
8: Update the i^{th} robot's current state s_i using $\boldsymbol{\omega}_i$
9: $\boldsymbol{\mathcal{I}}_i \leftarrow \mathcal{P}^m(\boldsymbol{\omega}_i, a_i)$
10: The i^{th} robot follows $\boldsymbol{\mathcal{I}}_i$ using a controller
11: **end if**
12: **end for**
13: $\lambda \leftarrow \mathcal{V}(\boldsymbol{\omega}, \boldsymbol{\mathcal{I}})$ ▷ Visualizer $\mathcal{V}$ defined in Section 4.2.4
14: Collect feedback, H, from human, where the human gives feedback based on task completion status and λ
15: $\boldsymbol{\mathcal{C}} \leftarrow \mathcal{R}(H)$ ▷ Restrictor $\mathcal{R}$ defined in Section 4.2.2
16: $\boldsymbol{P} \leftarrow \mathcal{P}^t(\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{G}}, \boldsymbol{\mathcal{C}})$ ▷ Update plans
17: **end while**

0 is false, meaning that all robots have an empty plan, i.e., all tasks have been completed. In each iteration, ARROCH enters a for-loop for updating the current configuration and the intended trajectories of the robots yet to reach their goal states (Line 4-12). The θ function in Line 7 returns the configuration of the i^{th} robot, which is stored at $\boldsymbol{\omega}_i$. In Line 9, $\mathcal{P}^m$ generates the intended trajectory of the i^{th} robot, $\boldsymbol{\mathcal{I}}_i$, to implement action a_i. After the for-loop, the set of robot configurations and robots' intended trajectories, are passed to **Visualizer** (Section 4.2.4), represented by the $\mathcal{V}$ function, which renders a single frame, λ, on the AR interface (Line 13).

The AR interface of ARROCH allows the human to give feedback on (the AR-based visualization of) the robots' plans. ARROCH obtains the human feedback, H (Line 14), and then passes it on to **Restrictor** (Line 15). Restrictor generates a set of activated constraints in logical form, $\boldsymbol{\mathcal{C}}$, that can be processed by Planner

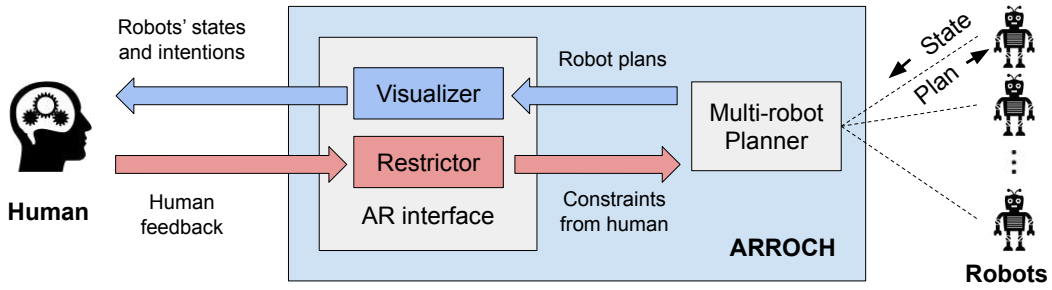


Figure 4.2: Key components of our ARROCH algorithm and system: *Visualizer* and *Restrictor* for visualizing robots' intention (for people) and collecting human feedback (for robots) respectively, and *Planner* for computing one action sequence for each robot.

in Line 16 (details in Sections 4.2.2 and 4.2.3).

Remarks: Leveraging a novel AR interface, ARROCH enables bidirectional, multi-turn, beyond-proximity communication toward collaborative behaviors within human-multi-robot teams. ARROCH is particularly useful in domains with poor visibility, e.g., multi-room indoor environments. The multi-robot task replanning capability enables the robots to collaborate with each other, while responding to human feedback by adjusting their task completion strategies.

Figure 4.2 shows an overview of ARROCH. In the following subsections, we describe how **Restrictor** converts human feedback into symbolic constraints, how **Planner** incorporates the constraints to plan for the robots, and how the robots' current and planned actions are visualized in **Visualizer** – all within our ARROCH system.

4.2.2 Restrictor for Constraint Generation

ARROCH realizes human-multi-robot collaboration by adding human-specified symbolic constraints into the multi-robot task planner. Within the task planning context, a **Constraint** is in the form of modal-logic expressions, which should

be true for the state-trajectory produced during the execution of a plan [148]. However, naive people have difficulties in directly encoding constraints in logical forms. ARROCH leverages the *graphical user interface* (GUI) of AR to take human feedback, with which Restrictor generates logical constraints. We use Answer Set Programming (ASP) for encoding constraints [5, 149]. A constraint in ASP is technically a STRIPS-style "headless" rule:

```
:- B.
```

where `B` is a conjunction of literals. It conveys the intuition of a constraint: satisfying `B` results in a contradiction. There is a rich literature on ASP-based logic programming that provides more technical details [150, 4].

We predefine a constraint library F, which is presented to people through the AR-based GUI. The human can select constraints $H \subseteq F$ to give feedback on the robots' plans (Line 14 in Algorithm 1). The form of human feedback can be flexible and task-dependent. For instance, in tasks that involve navigation, the human can forbid the robots from entering area `A` in step `I` using the following constraint,

```
:- in(A,I), area(A), step(I).
```

and in collaborative assembly tasks, the human can reserve a tool for a period of time. ARROCH uses Restrictor $\mathcal{R}$ to convert H into a set of constraints, $\boldsymbol{\mathcal{C}}$, which is the logical form of human feedback. Next, we describe how $\boldsymbol{\mathcal{C}}$ (from human) is incorporated into multi-robot task planning.

4.2.3 Multi-Robot Task Planner

In this subsection, we focus on the multi-robot task planner, $\mathcal{P}^t$ (referred to as **Planner**), that computes one task plan for each robot from the team of robots, while also considering the human feedback (Lines 2 and 16 in Algorithm 1). The input of $\mathcal{P}^t$ include, a set of robot initial states, $\boldsymbol{\mathcal{S}}$; a set of robot goal states, $\boldsymbol{\mathcal{G}}$; and a set of constrained resources, $\boldsymbol{\mathcal{C}}$.

Jointly planning for multiple agents to compute the optimal solution is NP-hard [151]. We adapt an *iterative inter-dependent planning* (IIDP) approach to compute joint plans for multiple robots [152]. IIDP starts with independently computing one plan for each robot toward completing their non-transferable tasks. After that, in each iteration, IIDP computes an optimal plan for a robot under the condition of other robots' current plans (i.e., this planning process depends on existing plans). While IIDP does not guarantee global optimality, it produces desirable trade-offs between plan quality and computational efficiency.

In the implementation of ARROCH, we use ASP to encode the action knowledge (for each robot) that includes the description of five actions: `approach`, `opendoor`, `gothrough`, `load`, and `unload`. For instance,

```
open(D,I+1) :- opendoor(D,I),
               door(D), step(I).
```

states that executing action `opendoor(D,I)` causes door `D` to be open at the next step. The following states that a robot cannot execute the `opendoor(D)` action, if it is not facing door `D` at step `I`. Such constraints are provided by Restrictor, as described in Section 4.2.2.

```
:- opendoor(D,I), not facing(D,I).
```

The output of $\mathcal{P}^t$ is $\boldsymbol{P}$, a set of task plans (one for each robot). For instance, $p_i \in \boldsymbol{P}$ can be in the form of:

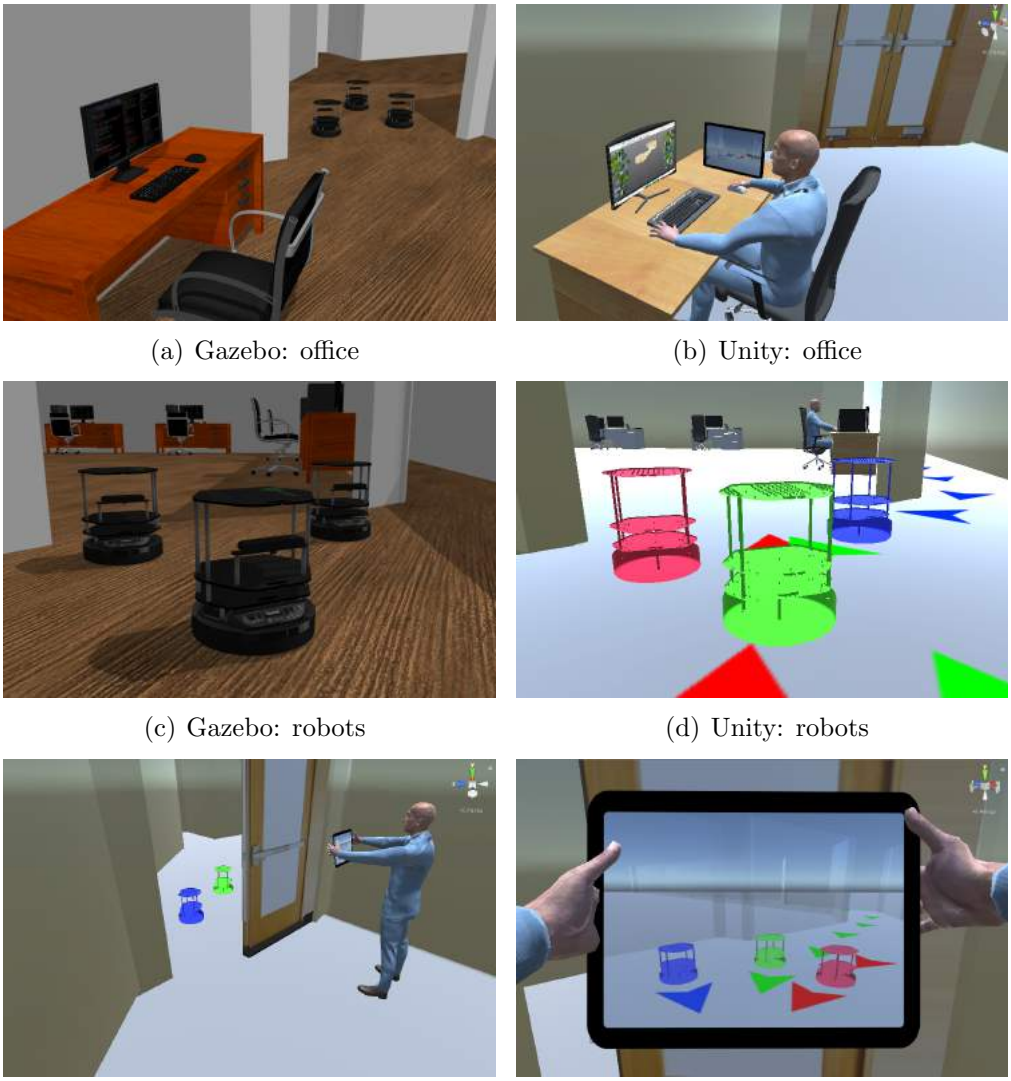
```
load(O,0).  approach(D,1).  opendoor(D,2).
gothrough(D,3).  unload(O,4).
```

suggesting the i^{th} robot to pick up object `O`, approach door `D`, enter a room through `D`, and drop off the object. Next, we describe how ARROCH handles the visualization of robots' current states and their planned actions through an AR interface.

4.2.4 AR Visualizer

ARROCH uses an AR interface to bridge the communication gap in human-multi-robot teams, where **Visualizer** $\mathcal{V}$ in Line 13 of Algorithm 1 plays a key role. The input of $\mathcal{V}$ includes the robots' current configurations in C-Space, $\boldsymbol{\omega}$, and their intended motion trajectories, $\boldsymbol{\mathcal{I}}$. The output of $\mathcal{V}$ are spatially visualizable 3D objects which are augmented over the real world. To accurately overlay virtual objects over the real world, AR devices need to localize itself in 3D environments. ARROCH assumes the availability of a set of landmark objects with known 3D poses (position and orientation). Using visual localization, ARROCH computes the pose of the AR device, and then accordingly augments visual information over the mobile device. We use a tablet for AR, though ARROCH is compatible with other mobile devices, e.g., head-mounted displays. Figure 4.1 (b) presents an example of the illustrative visualization.

In this section, we describe the ARROCH algorithm and system, the first contribution of this framework. Next, we delineate a novel simulation platform



(a) Gazebo: office

(b) Unity: office

(c) Gazebo: robots

(d) Unity: robots

(e) Unity: AR (3rd person POV)

(f) Unity: AR (1st person POV)

Figure 4.3: (a),(c), Gazebo environment for simulating multi-robot behaviors; (b),(d), Unity environment for simulating human behaviors and her AR-based interactions with the robots; and (e)-(f), Simulated AR interface.

(our **second contribution** of this framework) for prototyping and evaluating AR-based human-multi-robot collaboration algorithms.

SUGAR2: *S*imulation with *U*nity and *G*azebo for *A*ugmented *R*eality and *R*obots: Our simulation platform has been **open-sourced** under the name of **SUGAR2**.[2] SUGAR2 is built on Gazebo [122] for simulating robot behaviors, and Unity [146] for simulating human behaviors and AR-based interactions.

[2]https://github.com/kchanda2/SUGAR2

Figure 4.3 shows the Gazebo and Unity environments, including a simulated AR interface. Gazebo does not support the simulation of AR-based interactions, and Unity is weak in simulating robot behaviors, which motivated the development of SUGAR2.

4.3 Experiments

We have conducted experiments in simulation and using real robots. We aim to evaluate **two hypotheses**: I) ARROCH improves the overall efficiency in human-multi-robot team task completion, determined by the slowest agent of a team, in comparison to AR-based methods that do not support bidirectional human-robot communication; and II) ARROCH produces better experience for non-professional users in human-multi-robot collaboration tasks in comparison to non-AR methods. **Hypothesis-I** was evaluated in simulation and using real robots, whereas **Hypothesis-II** was only evaluated based on questionnaires from real human-robot teams.

In both simulation and real-world environments, a human-multi-robot team works on **collaborative delivery tasks**. The robots help people move objects from different places into a storage room, while the human helps the robots open a door that the robots cannot open by themselves. At the same time, the human has her own dummy task (solving Jigsaw puzzles). Tasks are non-transferable within the human-multi-robot team. The tasks and experiment settings are shared by simulation and real-world experiments.

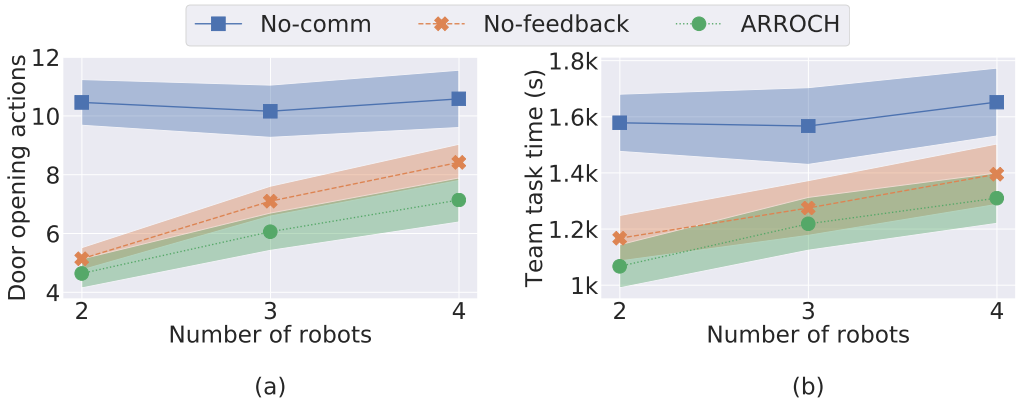


Figure 4.4: With different numbers of robots, ARROCH enables the human-robot team to: (a) reduce the human's workload, measured by the number of door-opening actions, and (b) improve the task-completion efficiency, at the same time. The total delivery workload (the number of objects to be delivered) is proportionally increased when the number of robots increases.

4.3.1 Simulation Experiments

We use the SUGAR2 environment (Section 4.2.4) to simulate human and robot behaviors, and their AR-based interactions. ARROCH has been compared with **two baselines**:

- **No-feedback:** It is the same as ARROCH, except that Lines 14-15 in Algorithm 1 are deactivated. As a result, the human can see the robots' current states and planned actions, but cannot provide feedback to the robots.
- **No-comm:** It is the same as No-feedback, except that Line 13 is deactivated. As a result, AR-based communication is completely disabled.

Three types of human behaviors are simulated in SUGAR2. a_0^H, *Work on her own task (simplified as sitting still in Unity)*; a_1^H, *Help robots open the door*; and $a_2^H(N)$, *Indicate unavailability in the next N minutes*.

When the AR-based visualization is activated (ARROCH and No-feedback),

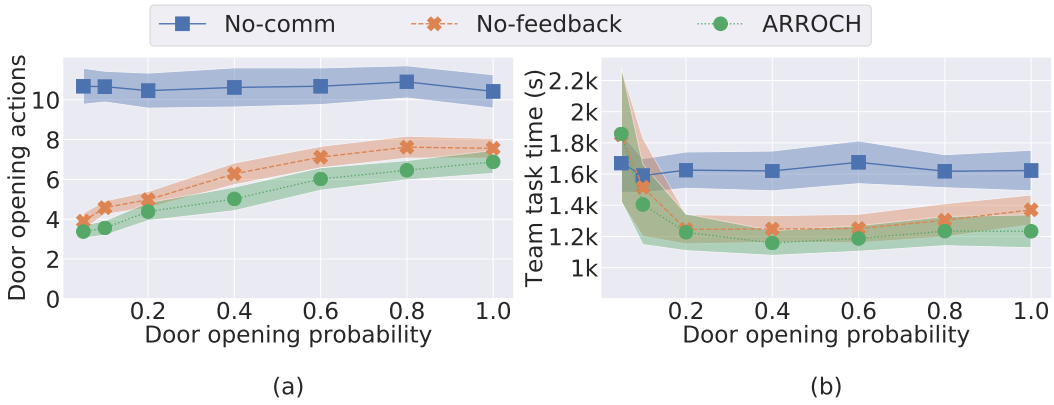
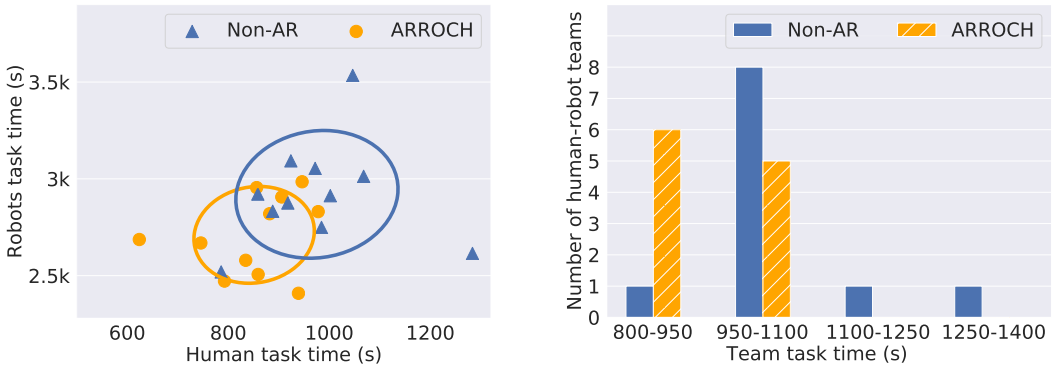


Figure 4.5: With different door-opening probabilities (the lower it is, the lazier the human is), ARROCH enables the team to: (a) reduce the number of door-opening actions, and (b) improve the task-completion efficiency, as long as the human is reasonably cooperative (i.e., willing to help the robots in ≥ 0.1 probability).

the human's door-opening behavior is independently triggered by each waiting robot, meaning that the human is more likely to open the door when more robots are waiting for the human to open the door. For No-comm, in every minute, there is a probability (0.6 in our case) that the human goes to open the door. To realistically simulate human behaviors, we added noise (Gaussian) into the human's task completion time. Before the human completes her task, we randomly sample time intervals during which the human will be completely focusing on her own task. At the beginning of such intervals, there is a 0.9 probability that the human takes action $a_2^H(N)$, i.e., indicating this "no interruption" time length to the robots in ARROCH.

Different Number of Robots: Figure 4.4 presents the results of simulation experiments, where we varied the number of robots in the human-multi-robot team. Each robot's workload is fixed. In our case, each robot needs to deliver three objects. Figure 4.4 (a) shows that ARROCH produced the lowest number of door-opening actions which directly reduces the human's workload. Figure 4.4 (b)



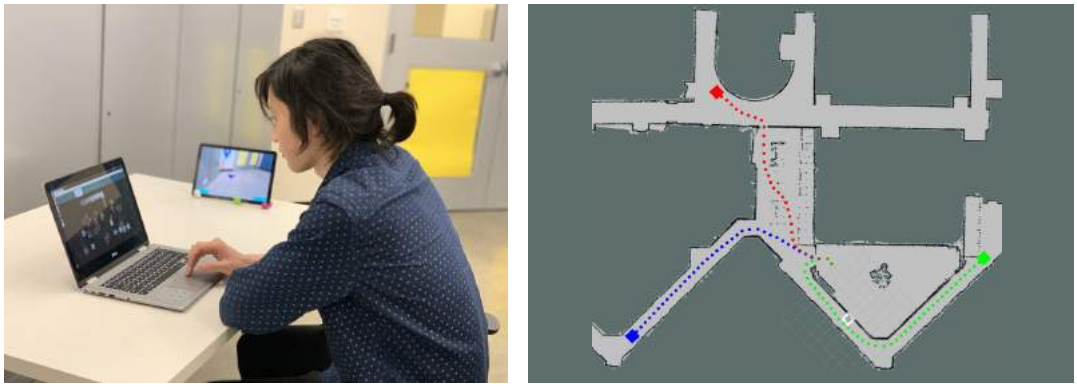
(a) Individual completion time

(b) Team completion time (histogram)

Figure 4.6: (a) ARROCH performed better than a traditional non-AR baseline (Rviz-based) in individual task completion time; and, (b) ARROCH performed better, c.f., non-AR, in team task completion time.

shows that ARROCH produced the best performance in the human-multi-robot team's task-completion time, which is determined by the slowest agent. The results support that Hypothesis-I is valid in teams of different sizes.

Human Laziness: Figure 4.5 shows the results of an experiment, where we evaluated ARROCH's performance under different human behaviors. We varied the human's laziness level which is inversely proportional to the probability that the human opens the door after she sees (through AR) a robot waiting outside. From the results, we see that ARROCH significantly reduced the number of door-opening actions (Left subfigure), while producing the best performance in task completion efficiency (Right subfigure), except for situations where the "door opening probability" is very low, meaning that the human is very lazy. This observation makes sense, because ARROCH only *encourages* lazy people to help robots (so does No-feedback) because the door opening behavior is triggered based on the number of visualized robots. On the other hand, "No-comm" forces people to open the door after every minute. The results support that Hypothesis-I is valid as long as the human is reasonably cooperative.



(a) Human participant (b) Non-AR baseline

Figure 4.7: (a) A human participant playing the Jigsaw game alongside our AR interface running on a tablet, while helping the robots open the yellow door, where simulation environment shown in Figure 4.3(b) was constructed accordingly; and (b) The Rviz-based interface (non-AR baseline) showing the three robots' current locations and planned trajectories.

4.3.2 Real-world Experiments

We have conducted experiments with human participants collaborating with multiple robots in the real world. The tasks to the human-robot team are the same as those in simulation: robots work on delivery tasks, and the human plays Jigsaw puzzles while helping robots through door-opening actions. The puzzles at the same difficulty level were randomly generated in each trial to avoid the participants' learning behaviors. We selected the Jigsaw game due to its similarity to assembly tasks in the manufacturing industry. The **real-world setup** is shown in Figure 4.7(a). We compared ARROCH to a standard **non-AR baseline**, where the human uses a Rviz-based interface to visualize the robots' current states and planned actions, as shown in Figure 4.7(b).[3]

Participants: Eleven participants of ages 20-30, all students from Binghamton University (BU), volunteered to participate in the experiment, including four females and seven males. Each participant conducted two trials using both the

[3]http://wiki.ros.org/rviz

Rviz-based baseline, and ARROCH – randomly ordered. None of the participants had any robotics background. There was no training performed on ARROCH or the baseline prior to the trials. The experiments were approved by the BU Institutional Review Board (IRB).

Task Completion Time: Figure 4.6(a) shows *individual task completion times* produced by ARROCH and the non-AR baseline. The x and y axes correspond to the human's task completion time, and the total of the individual robots' task completion times. The two ellipses show the 2D standard deviations. Results support Hypothesis-I which states ARROCH improves the human-robot team's task completion efficiency. To analyze the statistical significance, we sum up the team agents' individual completion times (both human and robots), and found that ARROCH performed **significantly better** than the non-AR baseline, with $0.01 < p\text{-value} < 0.05$.

We also looked into the *team task completion time*, which is determined by the time of the slowest agent (human or robot). Figure 4.6(b) shows the histogram, where the observation is consistent with that of individual task completion time. For instance, in most trials of ARROCH, the teams used ≤ 950 seconds, whereas, in most trials of non-AR, the teams could not complete the tasks within the same time frame. We observed that the participants, who did not have a robotics background, frequently experienced difficulties in understanding the visualization provided by the Rviz-based interface, including the map and robot trajectories. More importantly, ARROCH allows the participants to focus on their own tasks by indicating their unavailability, which is particularly useful in the early phase, whereas Rviz-based communication is unidirectional.

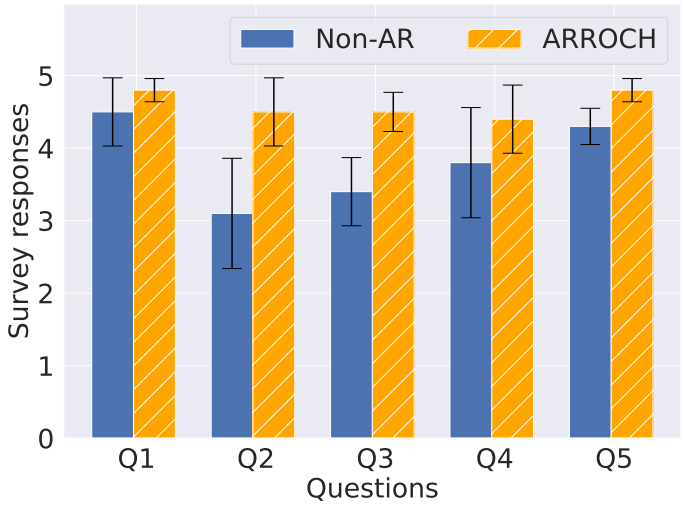


Figure 4.8: ARROCH produces a better user experience based on results collected from user questionnaires.

Questionnaires: At the end of each trial, participants were asked to fill out a survey form indicating their qualitative opinion on the following items. The response choices were: 1 (Strongly disagree), 2 (Somewhat disagree), 3 (Neutral), 4 (Somewhat agree), and 5 (Strongly agree). The questions include: Q1, *The tasks were easy to understand*; Q2, *It was easy to keep track of robot status*; Q3, *I could focus on my task with minimal distraction from robot*; Q4, *The task was not mentally demanding (e.g., remembering, deciding, and thinking)*; and Q5, *I enjoyed working with the robot and would like to use such a system in the future.* Among the questions, Q1 is a verification question to evaluate if the participants understood the tasks, and is not directly relevant to the evaluation of our hypotheses.

Figure 4.8 shows the average scores from the questionnaires. Results show that ARROCH produced higher scores on Questions Q2-Q5, where we observed **significant improvements** in scores in Q2, Q3, and Q5 with the p-values $<$ 0.001. The significant improvements support Hypothesis-II on user experience: ARROCH helps keep track of the robot's status, is less distracting, and is more

user-friendly. The improvement in Q4 was not significant, and one possible reason is that making quantitative comparisons over the "mentally demanding" level can be difficult for the participants.

4.4 Summary

Leveraging AR technologies, we introduce *AR for robots collaborating with a human* (ARROCH), a novel algorithm and system that enables bidirectional, multi-turn, beyond-proximity communication to facilitate collaborative behaviors within human-multi-robot teams. ARROCH enables the human to visualize the robots' current states and their intentions (planned actions), while supporting feedback to the robots. The human feedback is then taken by the robots toward human-robot collaboration. Experiments with human participants showed that ARROCH performed better than a traditional non-AR approach, while simulation experiments highlighted the importance of ARROCH's bidirectional communication mechanism.

5 Learning Visualization Policies from Demonstrations

In this chapter, I discuss my contribution to employing imitation learning to learn a visualization policy for the Augmented Reality agent. I describe the framework we develop, called VARIL, that enables AR agents to learn visualization policies (what to visualize, when, and how) from demonstrations. This is another contribution which falls under the proximate communication type, where human and robots are collocated.

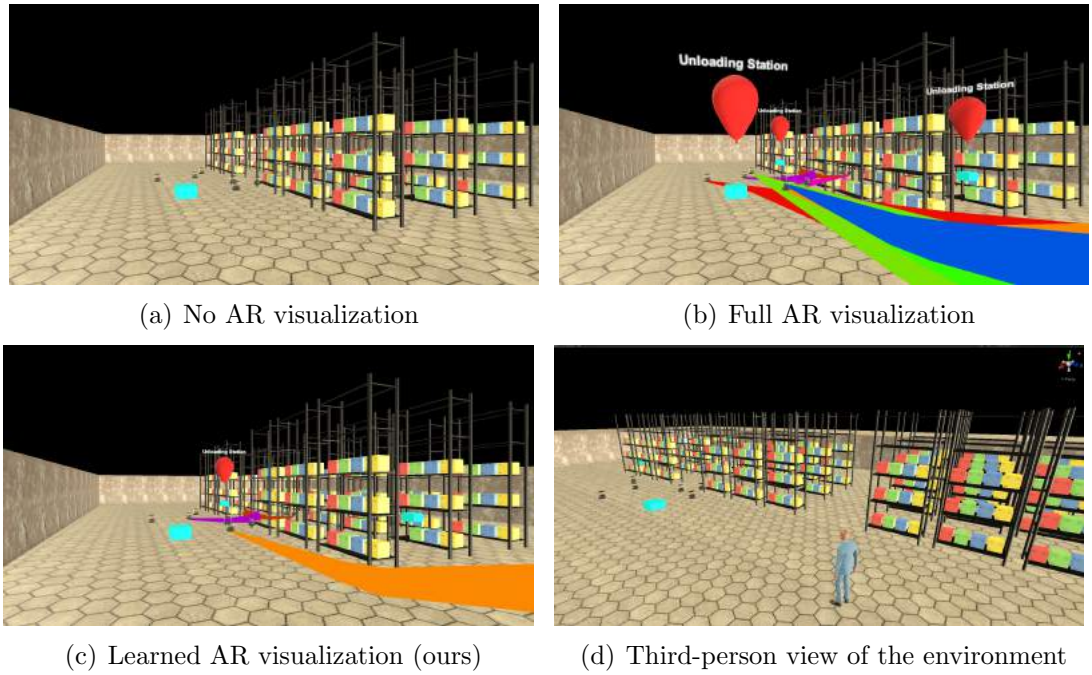
5.1 Introduction

Robots are increasingly being used in industrial environments, such as manufacturing and warehouses, where the workspaces of people and robots are usually isolated from each other. **Human-robot collaboration (HRC)** opens up a plethora of opportunities where people and robots could complement their capabilities. Hence, robots are entering human-inhabited environments to work with humans. Humans and robots prefer different communication modalities. While humans are used to natural language and gestures in communication, robots exchange information in digital forms, such as text-based commands, resulting in a communication gap in HRC. Researchers have developed algorithms and systems to bridge this communication gap using natural language [153, 18, 16, 154, 19]

and vision [20, 21, 22]. AR has been employed for HRC to provide an alternative communication mechanism with high bandwidth and low ambiguity [155].

AR is a technology that (1) combines real and virtual objects in a real environment; (2) runs interactively, and in real-time; and (3) registers (aligns) real and virtual objects with each other [143]. Such composite views can enhance human perception of the real world, and also result in an interactive experience of the environment. Many researchers are working on uncovering the benefits of AR in HRC [3, 155]. For instance, researchers have developed AR systems that allow humans to visualize the state of the robots [70], as well as the robot intentions [24, 27]. Existing AR-based HRC systems employ a static visualization policy where the visualizations are always displayed to the user. The handcrafted static policies might suggest different visualization strategies, but those systems are not able to take important runtime factors into account (e.g., the number of robots, the current status of robots, the future intentions of robots, and the current status of human) to adapt the visualizations. The consequence is that those AR interfaces can alter people's attentional focus and result in *inattentional blindness* [156], if there is visual clutter caused by too many unwanted visualizations. Such concerns motivate our research on enabling our AR agent to learn a policy for dynamically selecting visualization actions given the state of the human-multi-robot system [157].

We design a framework called, ***Visualizations for Augmented Reality using Imitation Learning*** **(VARIL)**, for human-multi-robot collaboration in a shared environment. VARIL allows human to track the status of a team of robots using an AR interface. In addition, VARIL learns the AR visualization



(a) No AR visualization

(b) Full AR visualization

(c) Learned AR visualization (ours)

(d) Third-person view of the environment

Figure 5.1: Different AR visualization strategies in a virtual warehouse environment, where a virtual human works with a team of mobile robots on collaborative delivery tasks. (a) No AR visualization, where the human worker does not know where some robots are, and what the robots plan to do. (b) Full AR visualization, where the human worker can be overwhelmed by the visual indicators. (c) Our learned AR visualization, where the AR agent uses a learned policy to dynamically determine a visualization strategy based on the current world state of both human and robots. (d) Third-person view of the human working alongside robots in the warehouse environment.

policies using **Imitation Learning (IL)** [158] that dynamically selects AR visualization actions for human-multi-robot teams. To the best of our knowledge, this is the first work that employs learning to design visualization strategies of AR for human-multi-robot collaboration. In particular, we train decentralized visualization policies in a centralized fashion [159]. While training, all the states are extracted from the agents in a centralized manner, and a new policy is generated by aggregating all the states, similar to learning in a single-agent environment. The new policy learned on the aggregated dataset is then deployed to the agents in a decentralized manner, where the agents query the actions from the joint policy, by using only the current agent's state information.

We empirically evaluated our VARIL framework in a warehouse domain [160, 161], where both the human and the team of robots are engaged in a collaborative delivery task. Figure 5.1 shows our warehouse environment where twelve turtlebots collaborate with a human. Both the human and robots are assigned nontransferable tasks, where the robots are tasked with delivering objects to different locations (drop stations) and the human helps unload the objects from the robots waiting at drop stations. We collected a dataset of 6000 state-action pairs using the feedback provided by the "expert demonstrator". We have trained the AR agent in the simulated warehouse environment with this dataset using an iterative IL algorithm [162]. The learned AR visualization policies were compared with the static AR policy for human multi-robot teams from the literature [163, 164], and the results suggest that the conflicts between the expert actions and the VARIL policies drastically decrease with the increase in the number of policy iterations. Additionally, we compared the total robot wait times for the intermediate policies of VARIL, and observed a significant decrease in the robot wait time between the initial and the learned AR visualization policies of VARIL.

While we observed significant improvements in the human-robot collaboration efficiency in pure simulation, we decided to evaluate how VARIL influenced the user experience. We have evaluated VARIL with 25 participants where every participant operated a virtual human in the warehouse environment. Every participant was asked to fill out a survey questionnaire that was used for subjective evaluations. Based on the participants' responses, we observed that our learned visualization strategy significantly reduced the distraction caused by extraneous visualizations. In addition to the qualitative evaluation, we also compared the

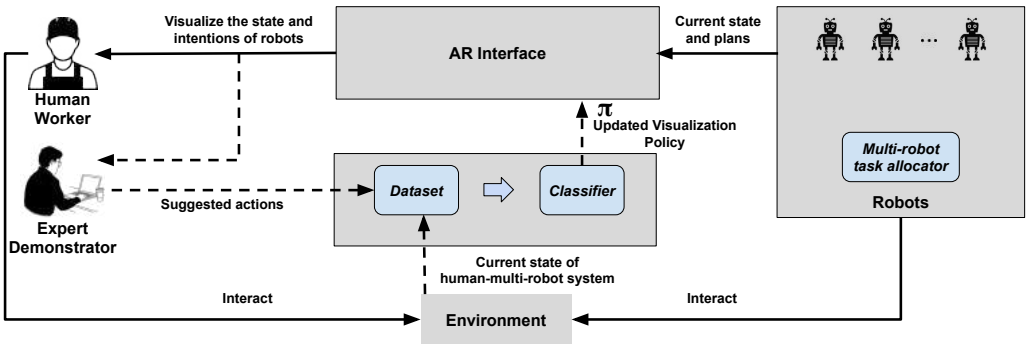


Figure 5.2: Overview of our VARIL framework that enables the human-multi-robot teams to collaborate in a shared environment. VARIL allows human to track the status of a team of robots using an AR interface. Under the hood, VARIL supports learning an AR visualization policy using expert demonstrations, where the learned policy dynamically selects the actions for the AR agent.

task completion times during the human study. The results show significant improvement in the efficiency of human-multi-robot teams in task completion as compared to two competitive baselines from the AR-for-HRC literature [65, 163]. Finally, we present a demonstration of a human worker collaborating with three mobile robots with AR-mediated communication in a built-in-lab mock warehouse environment.

5.2 Framework

In this section, we describe our framework called ***Visualizations for Augmented Reality using Imitation Learning*** **(VARIL)**, and how it enables human-multi-robot teams to collaborate together in a shared environment. VARIL allows human to track the status of a team of robots using an AR interface. Moreover, VARIL supports learning a decentralized AR visualization policy in a centralized fashion using expert demonstrations, where the AR agent uses the learned policy to dynamically select a visualization action based on the state of

the human-multi-robot teams.[1] Figure 5.2 showcases the VARIL framework, and the interplay between the different components of VARIL.

Consider a team of robots working with a human worker in a shared space. The team of robots constantly shares their state and plans using the AR interface, which is used by the human worker to track the status of the team of robots. The human worker simultaneously collaborates with a team of robots to complete the tasks. VARIL also consists of a human expert that gives demonstrations of AR visualizations at runtime, indicating what information should be visualized (or not) at specific times. Once a new policy is generated, the AR agent updates the visualizations to mimic the actions suggested by the expert demonstrator. There are two different human entities in VARIL, one for the role of the human worker, and the other is a human expert (Figure 5.2). Note that the expert is involved only during the policy learning phase, and once a satisfactory policy is learned, the expert demonstrator is no longer needed.

5.2.1 VARIL Algorithm

We design a human-multi-robot collaboration system using VARIL (Algorithm 2) that enables humans to work with a team of robots while using an AR interface to track the status of the robot teammates. The inputs to the algorithm are as follows:

- $\mathcal{P}^t$: Task pool for the robots
- N: Number of robots
- $\mathcal{P}^m$: Motion planner

[1] Our "centralized training, decentralized execution" idea was inspired by recent multiagent reinforcement learning research [159], while it is applied to an IL domain in this work.

Algorithm 2 VARIL

Require: $\mathcal{P}^t, \mathcal{T}^a, \mathcal{P}^m, N, \hat{\pi}^{latest}$

1: Initialize $\hat{\pi}^{latest}$ to show all visualizations
2: Initialize empty lists: $\boldsymbol{\omega} \leftarrow \emptyset$; $\boldsymbol{\mathcal{I}} \leftarrow \emptyset$
3: Initialize $\mathcal{S}^{\mathcal{R}} \leftarrow \emptyset$, and $\mathcal{H}$ to *null*, where $\mathcal{S}^{\mathcal{R}}$ is the state of N robots, and $\mathcal{H}$ is the state of the human worker
4: $\boldsymbol{T} = \mathcal{T}^a(\mathcal{P}^t, N)$, where $t \in \boldsymbol{T}$ is a task assigned to one robot, and $|\boldsymbol{T}| = N$ ▷ More details on $\mathcal{T}^a$ in supplementary materials.
5: **while** $\mathcal{P}^t$ is **not** empty **do** ▷ Tasks still exist in task pool
6: **for** $i \in [0, 1, \cdots, N\text{-}1]$ **do**
7: **if** t_i is **not** complete **then**
8: Obtain i^{th} robot's configuration: $\boldsymbol{\omega}_i \leftarrow \theta(i)$
9: $\boldsymbol{\mathcal{I}}_i \leftarrow \mathcal{P}^m(\boldsymbol{\omega}_i, t_i)$
10: The i^{th} robot follows $\boldsymbol{\mathcal{I}}_i$ using a controller
11: **else**
12: Update t_i using $\mathcal{T}^a$
13: **end if**
14: **end for**
15: $\lambda \leftarrow \mathcal{V}(\boldsymbol{\omega}, \boldsymbol{\mathcal{I}}, \hat{\pi}^{latest})$ ▷ More details on $\mathcal{V}$ in supplementary materials.
16: $\mathcal{S}^{\mathcal{R}} \leftarrow$ obtain all robots current state
17: $\mathcal{H} \leftarrow$ obtain current human worker state
18: $\hat{\pi}^{latest} \leftarrow$**PolicyUp**$(\mathcal{S}^{\mathcal{R}}, \mathcal{H})$ ▷ Update visualization policy
19: **end while**

- $\mathcal{T}^a$: Multi-robot task allocator that assigns tasks at runtime

Now, we describe in detail how our VARIL algorithm works. In Line 1, VARIL initializes $\hat{\pi}^{latest}$ to **enable all the visualizations**. Here visualizations represent all the 3D objects that are rendered in the AR interface to visualize the status of the team of robots. The intuition behind enabling all the visualizations is to enable the expert to easily identify different types of available visualizations. [2] The expert demonstrator can provide feedback on which visualizations are helpful, and which ones need to be turned on/off based on the observed state of the human-multi-robot system.

In Line 4, VARIL uses the multi-robot task allocator ($\mathcal{T}^a$) to allocate tasks

[2]This is to give the human expert full observability over the robots' current states and plans. If we had turned off the AR visualizations, there would be the issues, such as the human expert could not locate the robots obscured by other objects.

(T) for N robots, where T_r is the task of the r^{th} robot. Then, VARIL enters the main while-loop in Line 5 that runs until all the tasks from the task pool are completed. Inside the while-loop, VARIL enters a for-loop that runs for N times, once for each robot. At every iteration i of the for-loop, VARIL first checks if the robot has completed the current task or not (Line 7). If the task is not completed, VARIL first obtains the current robot configuration ($\boldsymbol{\omega}_i$). Then, VARIL uses $\mathcal{P}^m$ to generate the intended trajectory of the i^{th} robot, $\boldsymbol{\mathcal{I}}_i$. Once, the current task is complete, the robot obtains its new task using $\mathcal{T}^a$.

After the for-loop, in Line 15, the set of robot configurations, robots' intended trajectories, and the latest visualization policy is passed to the AR agent that renders the visualizations using the AR interface. In Lines 16 and 17, VARIL obtains the current state of all robots ($\mathcal{S}^{\mathcal{R}}$), and the current state of the human worker ($\mathcal{H}$). Finally, VARIL calls the PolicyUp function that updates the global policy $\hat{\pi}^{latest}$. This enables VARIL's AR agent to update the visualizations based on $\hat{\pi}^{latest}$ in the next iteration. In the next subsection, we explain how the PolicyUp function updates the visualization policy using IL based on the state of the human-multi-robot system and the expert demonstrations.

5.2.2 AR Policy Learning: PolicyUp

The input of Algorithm 3 includes, $\mathcal{S}^{\mathcal{R}}$, which is the state of robots; $\mathcal{H}$, the state of the human worker; $\mathcal{A}$, a set of agents for which the visualization is being learned; $\mathcal{J}$, the number of iterations for updating the policy; a dataset, $\mathcal{D}$, that stores the pairs of visited states and their corresponding actions suggested by the expert; and, $\hat{\pi}^{latest}$ that stores the latest visualization policy learned in $\mathcal{D}$.

PolicyUp enters the main for-loop in Line 1 that runs for $\mathcal{J}$ *times*. The value

Algorithm 3 PolicyUp($\mathcal{S}^{\mathcal{R}}, \mathcal{H}$)

Require: $\mathcal{J}$, $\mathcal{A}$, $\mathcal{D}$, $\hat{\pi}^{latest}$

1: **for** $j = 0$ to $\mathcal{J}$ - 1 **do**
2: **for** each agent α in $\mathcal{A}$ **do**
3: $\hat{\pi}_j \leftarrow \hat{\pi}^{latest}$ ▷ Latest visualization policy is used in the current iteration.
4: Let $\pi_j = \beta_j \pi^* + (1 - \beta_j)\hat{\pi}_j$
5: Sample T-step trajectories using π_j
6: Obtain expert feedback f on the observed states
7: Update visualization for agent α based on π_j
8: Get dataset $\mathcal{D}_j^{\alpha}$ = state of human-multi-robot system $(\mathcal{S}^{\mathcal{R}}, \mathcal{H})$, and expert suggested feedback (actions)
9: **end for**
10: Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_j^{\mathcal{A}}$ ▷ Combine individual datasets from all agents as one dataset
11: Train classifier $\hat{\pi}_{j+1}$ on $\mathcal{D}$ ▷ Common updated policy generated for all agents
12: Update $\hat{\pi}^{latest}$ with $\hat{\pi}_{j+1}$ ▷ Global policy is updated for visualization
13: **end for**
14: **return** $\hat{\pi}^{latest}$

of $\mathcal{J}$ can be customized based on the complexity of the state space of the AR agent. In each iteration, VARIL enters another for-loop in Line 2 that runs once for each agent, α, where $\alpha \in \mathcal{A}$. Inside the inner for-loop, PolicyUp first gets a policy π_j based on a decision rule, which decides if the expert policy or the novice policy will be used for the current iteration. The decision rule uses the value of β that decays over the iterations, and at the beginning, the value of $\beta = 1$, and hence expert policy actions are preferred over novice policy actions. As the value of j becomes closer to $\mathcal{J}$, the value of β becomes closer to 0. Intuitively this makes sense because at the beginning the novice policy is not mature enough and can lead to many mistakes, and hence the expert policy is used at the beginning. Then using the policy π_j, VARIL samples T-step trajectories in Line 5. During this phase, the visualizations are constantly updated for every agent α based on the current policy π_j (Line 7). Finally, for every agent, VARIL stores the visited

states as well as the suggested actions by the expert in $\mathcal{D}_j^{\alpha}$, where j represents the j^{th} iteration of the algorithm.

After completion of one iteration of the inner for-loop, the algorithm reaches Line 10 for data aggregation. Here, VARIL combines all the datasets stored in the current iteration ($\mathcal{D}_j^{\mathcal{A}}$) with the dataset from the previous iteration ($\mathcal{D}$). Note that even though the datasets were obtained from different agents, the data aggregation step considers them as they were obtained from a single agent. Such abstraction of data allows the algorithm to generate a comprehensive dataset based on the experiences of all the agents, and also enables VARIL to **train the visualization policies in a centralized manner with fully decentralized execution**. Once the dataset is aggregated, VARIL trains a classifier $\hat{\pi_{j+1}}$ on $\mathcal{D}$ in Line 11. The global policy $\hat{\pi}^{latest}$ is updated with the newly generated policy, which is then used by all the agents in the next iteration of the algorithm.

5.3 System Instantiation

In this section, we describe our implementation of the VARIL framework in a simulated warehouse environment (Figure 5.3). A key part of VARIL is an AR agent that learns to help a human worker visualize the status of a multi-robot team toward effective human-robot collaboration.

5.3.1 Warehouse Environment

We developed a platform for simulating warehouse environments using Unity [146], where human-robot teammates work together to complete delivery tasks. Figure 5.3 shows the simulated warehouse environment. The warehouse environment comprises of the following components:

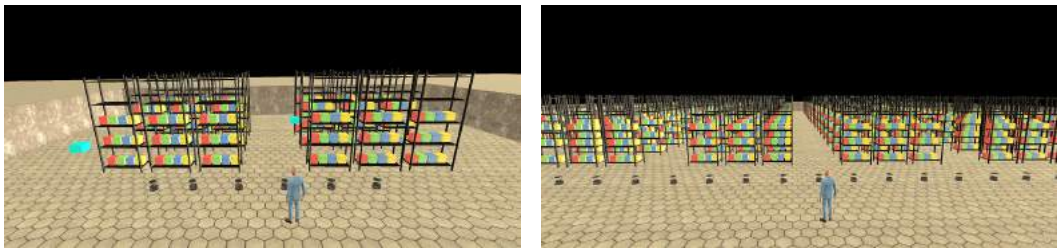
- A warehouse room
- Shelves with boxes
- A virtual human
- Drop stations
- Turtlebot robots

The robots and human work together in the shared warehouse room to complete collaborative delivery tasks. The robots move to different shelves to pick up boxes based on their tasks. After picking up the boxes, the robots move to the drop stations to deliver the objects and wait there until they get help from the human to unload the box.

We have created a virtual human in Unity that can be teleoperated by a real human to collaborate with the robots. While the robots are waiting at the drop station, the virtual human can be moved to a drop station to help the robots unload the objects. For simplicity, when the virtual human is in a radius of one meter from the robot waiting at the drop station, the robot automatically drops the box.

Additionally, our warehouse environment can be dynamically scaled by changing the parameters based on the desired complexity of the environment. Figure 5.3 shows an example of the two different scales of the warehouse environments. With the ability to choose the size of the environment, we can vary the complexity of the human-robot tasks.

Multi-robot Task Allocator ($\mathcal{T}^a$): We designed a multi-robot task allocator that assigns tasks to the team of robots at runtime. To allocate the tasks, $\mathcal{T}^a$ uses a task pool ($\mathcal{P}^t$) that contains all the tasks available for the robots. We use a random seed to initialize a pseudo-random number generator that is used to select the index of the task to be allocated to a particular robot.



(a) (b)

Figure 5.3: (a) A miniature version of the warehouse environment with 18 shelves and 6 Turtlebot robots. (b) A larger version of the warehouse environment with 225 shelves and 15 Turtlebot robots.

Motion Planner ($\mathcal{P}^m$): We use Unity NavMesh (short for Navigation Mesh) [165] for robot navigation and obstacle avoidance. We first generate a NavMesh, which is a data structure to describe the navigable surfaces of the environment, and allows the robot to find a path from one navigable location to another in the environment. Every robot is considered a NavMesh agent in Unity, and every NavMesh agent can avoid each other while moving towards their goal. All the agents in Unity reason about the environment using the NavMesh, and they know how to avoid each other as well as moving obstacles.

5.3.2 AR Agent for the AR Interface

In our instantiation of the VARIL system, we used four different types of visualizations in the AR interface to enable the human to track the robots' status in the warehouse environment. The different visualizations are as follows:

- **Trajectory** shows the motion intentions of the robot. Additionally, the width of the trajectory increases with the distance of the goal from that point on the trajectory. Due to the presence of multiple robots in the environment, the trajectories of different robots are overlaid using different colors so that the human can distinctly identify different robot plans.

- **Solid Avatar** shows the live location of the robot. As the robot moves toward the goal location, the position of the solid avatar is updated accordingly.

- **Transparent Robot Avatar** shows the direction of the motion of the robot by moving the avatar constantly over the trajectory.

- **Floating Balloon** enables the human to locate the drop station from a long distance.

5.3.3 State-Action Space of AR Agent

We learn a visualization strategy for two types of visualizations, one for the visualization of robots, and the other for the visualization of the drop station. The state-space of our AR agent for the robot consists of the following:

- humanState: {close, moderate, far}

- robotTaskState: {picking, dropping}

- robotRemainingTasks: {few, many}

- robotWaitingTime: {short, medium, long}

- nearbyRobots: {few, many}

- nearbyRobotVizStatus: {few, many}

The above state representation consists of all the key states that represent the entire human-multi-robot system from each agent's perspective. The action space of the AR agent depends on the number of different visualizations. In our case for robots, we have three different visualizations, which are:

- liveLocation: {on, off}
- transparentAvatar: {on, off}
- trajectory: {on, off}

where each of them can be turned on or off. The state-space of our AR agent for each drop station consists of the following:

- humanState: {close, moderate, far}
- robotsWaitingTimeDS: {short, medium, long}
- nRobotsAtDropStation: {few, many}

The action space for each drop station consists of enabling or disabling the balloon on the drop station, hence it is represented as follows:

- balloon: {on, off}

The above state and action space represent our implementation of the AR agent, and we use the PolicyUp function (Algorithm 3), which is a part of the VARIL framework to learn a visualization policy for human-multi-robot teams.

The next section provides details on the experiments conducted to evaluate VARIL in the human-multi-robot collaboration scenario.

5.4 Experiments

We conducted experiments in a simulated warehouse environment that we developed using Unity. With the experiments, we aim to evaluate **two hypotheses**:

I) VARIL improves the overall efficiency in human-multi-robot team task completion, determined by the slowest agent of a team, in comparison to other AR-based methods from the literature that employ static visualization policies; and II) VARIL provides a better user experience for humans in human-multi-robot collaboration tasks compared to competitive AR-based methods from the literature. We have evaluated both Hypothesis-I and Hypothesis-II with human participants collaborating with multiple robots in the simulated warehouse environment.

Participants: Twenty-five participants of ages 20-30, participated in the experiment, including six females and nineteen males. Every participant was given a $10 gift card for participating in the experiments. Each participant conducted three trials using three methods, including VARIL, that were randomly ordered for each participant. Before starting the actual trials, each participant completed a short dummy trial that was scaled down to a mini-warehouse that includes only one robot. The role of the dummy trial was to acquaint the participants with the environment and to teach the participant to control the virtual human. The experiments were approved by the Institutional Review Board (IRB) of our institution.

5.4.1 Experiment Setup

We deployed the simulated warehouse environment to a web server to facilitate the online experiment using the WebGL build of the Unity project. In our experiments, the human participants were able to teleoperate the virtual human around the environment through the input of the *W, A, S,* and *D keys* along with using the mouse to rotate the virtual human's field of view. Each robot is assigned three boxes for each trial and once the last box has been dropped off by the robot, it will

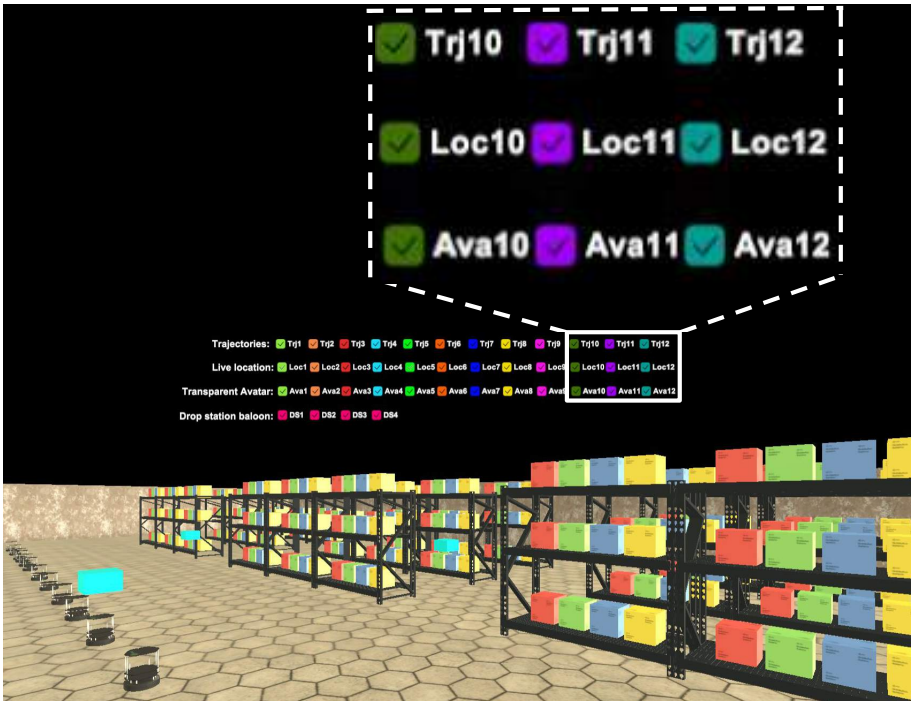


Figure 5.4: Interface used by the expert to give demonstrations.

begin navigating to its starting position. The trial is completed once all twelve robots have delivered their three boxes and have reached their starting position.

Data Collection and Policy Learning: We collected a dataset of 6000 demonstrations provided by a human expert at runtime during which a human worker was working with a team of robots to collaborate on a delivery task. In our case, the human expert is the author of this dissertation, who knows this domain well, and we designed a virtual human to perform the task of the human worker. The human worker and the human expert both were provided with an AR interface to track the robots. The human worker used the AR interface to collaborate with the robots, whereas the role of the human expert was to provide feedback on the visualizations seen using the AR interface.

The human expert was provided with a simple interface as shown in Figure 5.4 to input the feedback (which visualizations the human expert thinks should be to enabled or disabled) on the visualizations for the team of robots, as well as the drop

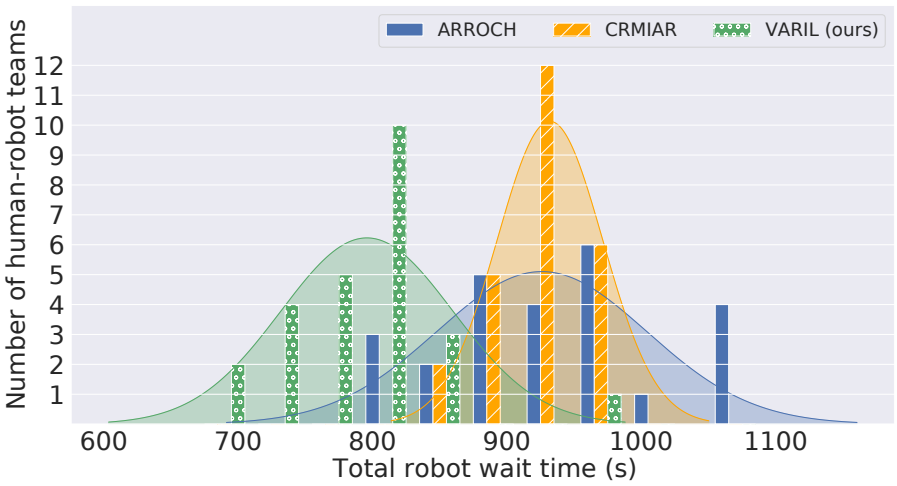
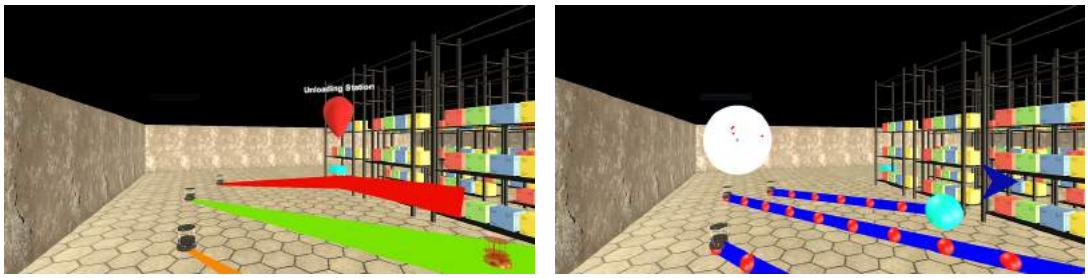


Figure 5.5: Comparison of VARIL (ours) with two baselines to show the distribution of total waiting time (second) of all robots.

station balloons. The interface provides a checkbox for each robot's trajectory, live location, and transparent avatar as well as a checkbox for each drop station balloon. The expert feedback is obtained in real-time and the environment is not paused. The expert does not have access to the internal state representation, and the feedback is completely based on the human expert's observations on the environment. Along with the state of the human worker and the team of robots, the feedback (actions) of the human was combined to collect the dataset. A snapshot of the state-action pairs was extracted every four seconds, and after every 25 state-action pairs, the dataset was aggregated with the dataset from the previous iteration. Using VARIL, we ran a total of two hundred forty iterations, creating the entire dataset of the state-action pairs. At every iteration, we train the AR agent with a multi-class SVM [166].

Baselines: For baselines, we have replicated the visualizations of two different systems from the literature, called ARROCH [163], and CRMIAR [65] [3] to make comparisons of their visualization strategies with the learned visualization from

[3]We create the acronym of CRMIAR that indicates "Communicating Robot Motion Intent with Augmented Reality" for the simplicity of referring to the method.



(a) ARROCH (b) CRMIAR

Figure 5.6: Our implementation of two existing methods from literature, ARROCH [163] and CRMIAR [65], showing their AR visualizations for enabling the humans to track the status of robots.

VARIL. ARROCH was designed for human-multi-robot systems, where a human can use an AR device to track the status of the robots. Our implementation of the ARROCH system has the following visualizations: the trajectories to show the motion intentions, the robot avatar to show the live location, and a transparent robot avatar to show the direction of the motion of the robot. On the other hand, CRMIAR was a single robot visualization system, but we have scaled it to enable tracking the team of robots in our warehouse environment. In our implementation of the CRMIAR system, where we have combined the NavPoints, and Utilities designs of CRMIAR to provide the following visualizations: the trajectories of robots to show their planned motion intentions, a map to show the relative location of the robots with respect to the human, and arrows to point to the robots that are not in the field of view. We use both ARROCH and CRMIAR in our experiments to evaluate the efficacy of VARIL. Figure 5.6 shows our adapted implementation of the baselines.

5.4.2 Real-world Demonstration

For demonstration, a built-in-lab mock warehouse was constructed in a 2500 square-foot room. Three turtlebot robots were used to mimic the human-multi-

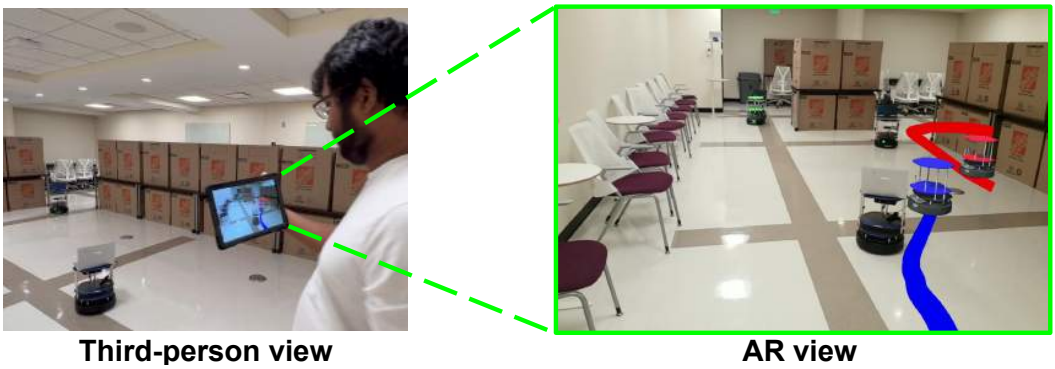


Figure 5.7: Mock warehouse and AR visualizations that include robot avatars and trajectories in color.

robot collaborative delivery task similar to the Unity environment. The speed of the Turtlebots was set to 0.5 m/s. The human worker was provided with an AR device (tablet) with a screen size of 10 inches. Figure 5.7 shows the "warehouse" environment for the demonstration of VARIL, where a human worker is holding an AR device to track the status of three turtlebots. A demonstration video is available on the project webpage – see the abstract.

5.4.3 Results

Objective Results: Figure 5.5 shows the histogram, where the x-axis represents the total wait time of all the robots in seconds, and the y-axis represents the number of human-robot teams with the corresponding robot wait times. From the figure, it can be observed that for most human-robot teams, VARIL had the shortest robot wait times as compared to ARROCH and CRMIAR. Also, we plotted the Gaussian curves to show the distribution of the data points for all the methods. We also analyzed the statistical significance, in every trial, we sum up the task completion time of all the robots. We found that VARIL performed **significantly better** than both ARROCH and CRMIAR, where $0.01 < p - value < 0.05$. Additionally, by comparing the total robot wait times, we observed that the wait

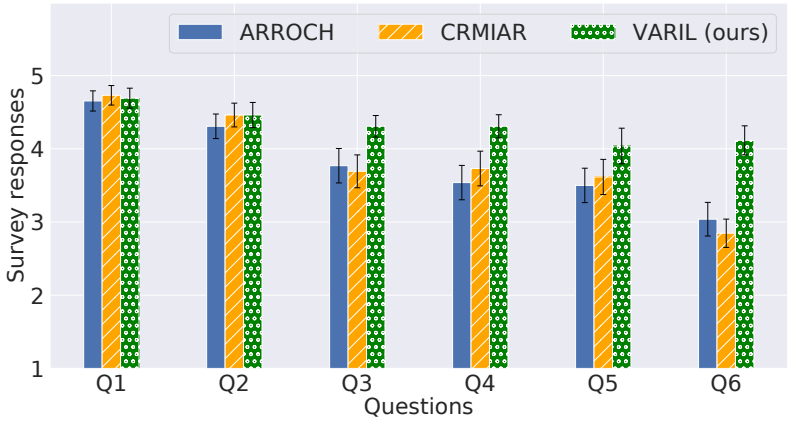


Figure 5.8: Questionnaire scores

times for robots in VARIL were significantly shorter than the baselines. All of these results support Hypothesis-I which states VARIL improves the human-robot team's task completion efficiency.

Subjective Results: At the end of each trial, the participants were asked to fill out a survey form indicating their qualitative opinion on the six different questions to validate Hypothesis-II. The figure and the descriptive subjective analysis are provided in the supplementary materials. We observed significant improvements from the study supporting Hypothesis-II on user experience, that the visualizations in VARIL provided a better user experience. Collectively, the results convey that the VARIL visualizations help the participant better keep track of the robot status, proved useful towards task completion, and were less distracting, while the participants enjoyed using the system.

Questionnaire for Subject Analysis: The questions listed in the questionnaire included: Q1, *The tasks were easy to understand*; Q2, *The tasks were not mentally demanding, e.g., remembering, deciding, thinking, etc.*; Q3, *I enjoyed using the system and would like to use such a system in the future.*; Q4, *It was easy to*

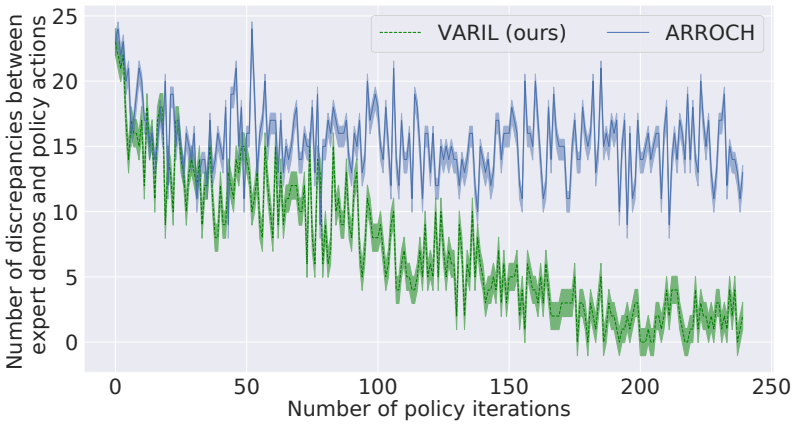


Figure 5.9: (a) The number of discrepancies between the expert demonstrator's suggested actions with both the visualization policies of ARROCH (baseline) and VARIL with respect to the number of iterations. Each instance of a human expert disabling a visualization is considered a **discrepancy**;

keep track of robot status.; Q5, *The visualizations proved helpful in completing the tasks.*; and Q6, *The visualizations were not distracting.* Note that Q1 is a verification question to evaluate if the participants understood the tasks, and is not directly relevant to the evaluation of our hypotheses. The response choices were: 1 (Strongly disagree), 2 (Somewhat disagree), 3 (Neutral), 4 (Somewhat agree), and 5 (Strongly agree).

Figure 5.8 (presented in the supplementary materials) shows the average scores from the questionnaires. Results show that VARIL produced higher scores on questions Q3-Q6, where we observed **significant improvements** in all of the four questions with the p-values < 0.05. The significant improvements observed from the study support Hypothesis-II on user experience that the visualizations in VARIL provided a better user experience. We did not observe a significant difference in scores for Q2, and one possible reason we thought is the way the question was framed. Since we ask about the demanding nature of tasks, and the response of Q1 confirms that the tasks were easily understood by the participants, the scores make sense. On the other hand, if the question was about the demanding

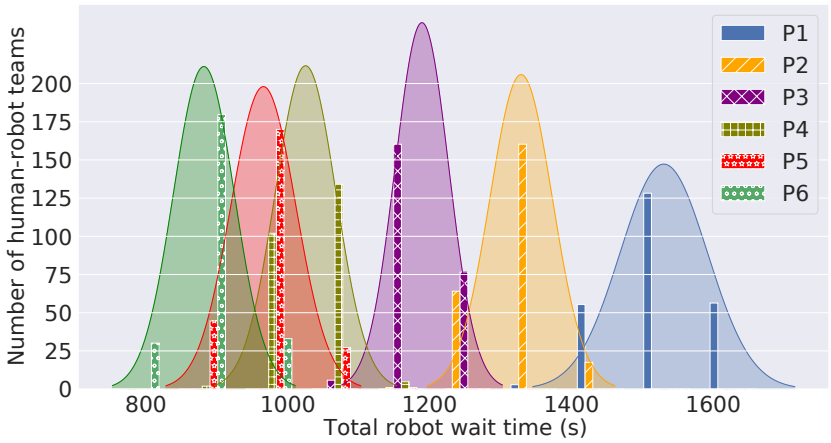


Figure 5.10: A histogram to show the distribution of total waiting time (second) of all robots in simulation with respect to six intermediate policies of VARIL.

nature of processing the visualization, we think the question would have been useful for the evaluation of different visualization strategies.

Policy Evaluation: VARIL is the first work on Imitation Learning-based visualization policy learning for AR, and hence we do not have learning baselines to make comparisons for the generated policy. To overcome this shortcoming, we have evaluated our learned policy to find the discrepancy between the suggested expert actions as well as the actions suggested by both the ARROCH static visualization policy, and the intermediate policies of VARIL. Figure 5.9 shows the number of policy iterations on the x-axis, and the number of discrepancies on the y-axis. The discrepancy is a measure of the deviation of the visualization policies from the expert's policy. From the figure, we can observe that the early policies of VARIL had more discrepancy, and this is because at the beginning the policy's poor quality makes it less effective in selecting visualization actions. With more interaction with the expert, the policy becomes mature, and tries to mimic the actions suggested by the expert. It can be observed that the discrepancies drastically decrease with the increase in the number of policy iterations. On the other

hand, the increase in policy iteration has no effect on the number of discrepancies between ARROCH and expert actions, because ARROCH employs a static visualization policy. The final policy was used in the experiments to evaluate the learned visualization policy against the static policies from the baselines.

Furthermore, we evaluated Hypothesis-I in simulation to investigate how the intermediate policies of VARIL affect the human-robot team's task completion time. Figure 5.10 shows the histogram of the total robot wait times, for six different policies (P1 - P6), where every policy was extracted after 50 policy iterations. A total of 250 trials were run for every policy, and we observed improvements in robot wait times denoting the improvement of efficiency in human-robot teams.

5.5 Summary

In this chapter, I present our framework, VARIL (*Visualizations for Augmented Reality using Imitation Learning*) [167], that for the first time introduces a learning-based AR visualization strategy for human-multi-robot collaboration. The framework learns a decentralized visualization policy using imitation learning in a centralized manner, where the AR agent dynamically selects a visualization action based on the state of the human-multi-robot system. One limitation of the work is that the author of the dissertation served as the expert demonstrator in the VARIL framework, a more diverse set of expert demonstrators may be beneficial for learning a more robust policy. We compared our framework with competitive baselines from the literature, and the results suggest that VARIL significantly increases the efficiency of human-robot collaboration, and reduces the distractions caused by extraneous information.

6 Learning to Guide Human Attention

In this chapter, I discuss my contribution to learning visualization policies for the telepresence systems for guiding human attention in all-degree vision. Our framework, GHAL360, enables the MTR to learn a goal-oriented policy from reinforcements for guiding human attention using AR visual indicators. This contribution falls under the remote communication type, where human and robots are not collocated, and the human and robot share the same embodiment.

6.1 Introduction

Telepresence is an illusion of spatial presence, at a place other than the true location [168]. Mobile telepresence robots (**MTRs**) enable the human operator to extend their perception capabilities along with the ability to move and actuate in a remote environment [169]. The rich literature of mobile telepresence robotics has demonstrated applications in domains such as offices [170, 171], academic conferences [172, 173], elderly care [174, 175], and education [176, 177].

Recently, researchers have equipped MTRs with a 360° camera to perceive the entire sphere of a remote environment [112, 111]. In comparison to traditional monocular cameras (including the pan-tilt ones), 360° visual sensors have equipped the MTRs with the capability of omnidirectional visual scene analysis. However,

the human vision system, by nature, is not developed for, and hence not good at processing 360° visual information. For instance, computer vision algorithms can be readily applied to visual inputs with varying spans of degrees, whereas the binocular visual field of the human eye spans only about 120° of arc [178]. The portion of the field that is effective for complex visual processing is even more limited, e.g., only 7° of visual angle for facial information [179]. *How can one leverage the MTRs' 360° visual analysis capability to improve the human perception of the remote environment?* One straightforward idea is to project 360° views onto equirectangular video frames (e.g., panorama frames). However, people tend to focus on small portions of equirectangular videos, rather than the full frame [180]. This can be seen in Google Street View, which provides only a relatively small field of view to the users, even though the 360° frames are readily available. Toward long-term shared autonomy, we aim to bridge the **observability gap** in human-MTR systems equipped with 360° vision.

In this chapter, we propose a framework, called Guiding Human Attention with Learning in 360° vision (**GHAL360**). To the best of our knowledge, GHAL360, for the first time, equips MTRs with simultaneous capabilities of 360° scene analysis and guiding human attention. We use an off-policy reinforcement learning (RL) method [14] to compute a policy for guiding human attention to areas of interest. The policies are learned toward enabling a human to efficiently and accurately locate a target object in a remote environment. More specifically, 360° visual scene analysis produces a set of detected objects, and their relative orientations; and the learned policies map the current world state, including both the human state (current attention), and the scene analysis output, to an action of AR visual

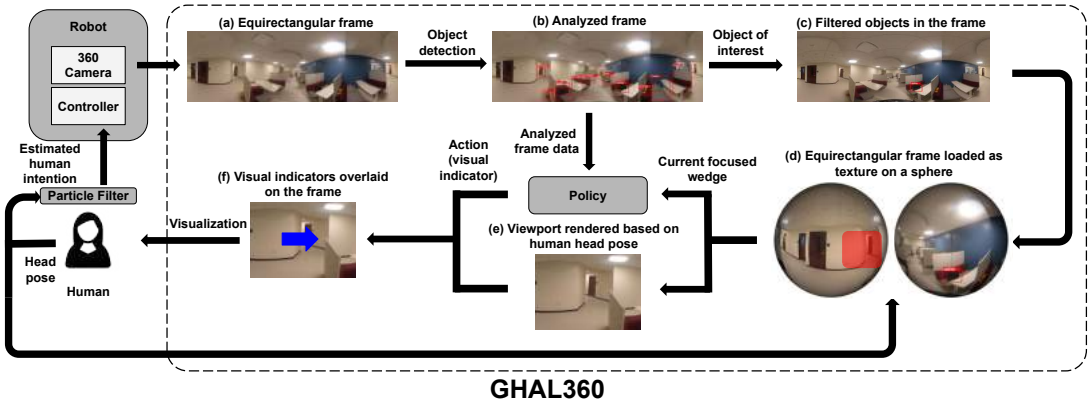


Figure 6.1: The input of GHAL360 includes live 360° video frames of the remote environment from MTR, and human head pose. (a) **Equirectangular frames** encode the 360° information using equirectangular projection. (b) Those frames are analyzed using an object detection system. (c) The detected objects are then filtered to highlight only the objects of interest, which is a task-dependent process. (d) After the human's display device receives the equirectangular frames, GHAL360 constructs a sphere, and uses the frames as the texture of the sphere. A human may only view a portion of the sphere, with a limited field of view, e.g., as indicated in the red-shaded portion. (e) The portion viewed by the human is called a "**viewport**", which is rendered based on the head pose of the human at runtime. (f) GHAL360 overlays AR visual indicators in real-time to guide human attention toward the object of interest while providing an immersive 360° view of the remote environment.

indication.

We have evaluated GHAL360 using target search tasks. Three virtual environments have been constructed for demonstration and evaluation purposes—two of them using the Matterport3D datasets [125], and one using a dataset collected from a mobile robot platform. We have compared GHAL360 with existing methods including [100, 110]. From the results, we see that GHAL360 significantly improved the efficiency of the human-MTR system in locating target objects in a remote environment.

6.2 Framework

In this section, we present our framework called GHAL360, short for Guiding Human Attention with Learning in 360° vision, that leverages the MTR's 360° scene analysis capability to guide human attention to the remote areas with potentially rich visual information. Figure 6.1 shows an overview of GHAL360.

6.2.1 GHAL360 Framework

Algorithm 4 delineates the procedure of GHAL360 and explains the different stages as well as the data flow at each stage. The input of GHAL360 includes κ, the name of a target object of interest as a string, and $objDet$, a real-time object detection system.

After a few variable initializations (Lines 1-4), GHAL360 enters the main control loop in Line 6. The main control loop continues until the human finds the target object in the remote environment. Once a new frame ($\mathcal{F}$) is obtained, GHAL360 analyzes the remote scene using an object detection system and stores the objects' names and their locations in a dictionary $\mathcal{P}$ (Line 9). The location of κ is then stored in $\mathcal{L}$ and is used to draw the bounding box over $\mathcal{F}$.

If the target object is detected in the current frame (Line 15), Lines 16-24 are activated for guiding human attention. In Line 16 the current egocentric state representation (s) of the world is obtained using the $getState$ function. Based on s, the policy generated from the RL approach (π) returns an action, which is a guidance indicator to be overlayed on the current viewport ($\mathcal{F}^{'}$), and then $F^{'}$ is presented to the user via the interface. After executing every action a, GHAL360 updates the Q-values for the state-action pair using the observed reward (r) and

Algorithm 4

Require: κ, $objDet$

1: Initialize an empty list $\boldsymbol{\mathcal{C}}$ to store the control commands as $\emptyset$
2: Initialize a quaternion $\mathcal{H}$ (Human Pose) as (0,0,0,0)
3: Initialize a 2D image $\mathcal{F}$ with all pixel values set as 0
4: Initialize a dictionary $\mathcal{P}$ ={}
5: $\pi \leftarrow$ random policy ▷ Initialize a policy
6: **while** True **do**
7: **if** new frame is available **then**
8: Obtain current frame $\mathcal{F}$ of the remote environment
9: $\mathcal{P} \leftarrow objDet(\mathcal{F})$ ▷ Section 6.2.1
10: $\mathcal{L} = \mathcal{P}[\kappa]$ ▷ Get the location of the object of interest
11: Overlay bounding box for object of interest over $\mathcal{F}$
12: Render the frame as a sphere ▷ Section 6.2.1
13: Obtain current human head pose $\mathcal{H}$
14: Render a *viewport* $\mathcal{F}^{'}$ to match the human operator's head pose
15: **if** κ in $\mathcal{P}$ **then**
16: $s = getState(\mathcal{P}, \kappa, \mathcal{H})$ ▷ Egocentric state representation
17: $a = \pi(s)$ ▷ Section 6.2.2
18: **if** $a == left$ or $right$ **then**
19: Overlay the visual indicator on $\mathcal{F}^{'}$
20: **end if**
21: Present $\mathcal{F}^{'}$ to the user via the interface
22: Observe r and s' ▷ Immediate reward and next state
23: $Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$
24: $\pi(s) \leftarrow argmax_a Q^*(s,a)$ ▷ Update policy
25: **end if**
26: $\mathcal{I} \leftarrow pf(\mathcal{H})$ ▷ Section 6.2.2
27: $\boldsymbol{\mathcal{C}} \leftarrow$controller($\mathcal{I}$) ▷ Control signal based on human intention
28: **for each** $c \in \boldsymbol{\mathcal{C}}$ **do** ▷ Section 6.2.3
29: $\psi \leftarrow \tau(c)$ ▷ Generate a teleop message
30: ψ is sent to the robot for execution
31: Updated map is presented via the interface
32: **end for**
33: **end if**
34: **end while**

the next state (s') in Line 23. Finally, using the new Q-values, the policy (π) is updated, and in the next iteration, an optimal action is selected using the updated policy. The human head orientation ($\mathcal{H}$) is sent to the particle filter as evidence to predict the human motion intention ($\mathcal{I}$). The controller then returns the control signal as $\boldsymbol{\mathcal{C}}$ based on $\mathcal{I}$ to control the robot's motion in the remote environment.

In the following subsections, we describe the key components of GHAL360.

Scene Analysis: Once a new frame ($\mathcal{F}$) is obtained, GHAL360 analyzes the remote scene information using an object detection system, where we use a state-

of-the-art convolutional neural network (CNN)-based framework **YOLOv3** [181]. All the objects (keys) in the frame along with their locations (values) are stored in a dictionary $\mathcal{P}$ (Line 9). In our implementation, we divide the 360° frame into eight equal 45° wedges. These eight wedges together represent all the four cardinal and four intercardinal directions, i.e. [N, S, E, W, NE, NW, SE, SW]. Every wedge can have four different values based on the objects detected in that wedge:

$$W : \{w_0, w_1, w_2, w_3\}$$

- w_0: represents a wedge with no objects detected
- w_1: represents a wedge with clutter
- w_2: represents a wedge that contains only the object of interest
- w_3: represents a wedge containing clutter as well as the object of interest

where clutter indicates that the wedge contains one or more objects that are not the target object. The output of scene analysis is stored as a vector where each element represents the status of one of the wedges. The human operator can be focusing on one of these eight wedges. The *getState* function in Line 16 takes the output of the scene analysis along with the target object and current human head pose as input. The output of the *getState* function is an egocentric state representation of the vector representing the locations of objects with respect to the wedge human is currently focusing on (W_0). The generated egocentric version of the vector is then sent to the RL agent as the state representation of the current world, including both the output of scene analysis and the current status of the human operator.

Equirectangular Frame Rendering: One of the ways to represent 360-degree videos is to project them onto a spherical surface. Consider the human head as a virtual camera inside the sphere. Such spherical projection helps to track the human head pose inside the 360° frames. In our implementation, we construct a sphere and use $\mathcal{F}$ as the texture information of the sphere. Inside the sphere, based on the yaw and pitch of the human head, the pixels in the human field of view are identified. Based on the human operator's head pose obtained in Line 13, the viewport is rendered from the sphere. We use **A-Frame**, an open-source web framework for building virtual reality experiences [1], for rendering the equirectangular frames to create an immersive 360° view of the remote environment.

6.2.2 Policy for Guiding Human Attention:

We use a reinforcement learning approach [14], Q-Learning, to let the agent learn a policy (for guiding human attention) by interacting with the environment. The mathematical representation of the state space is:

$$S : W_0 \times W_1 \times \cdots \times W_{N-1}$$

where W_i represents the state of a particular wedge, as defined in Section 6.2.1. In our case, $N = 8$, so $|S| = 4^8$, and $s \in S$. The state space (S) of the RL agent consists of a set of all possible state representations of the 360° frames. All the state representations are egocentric, meaning that they represent the state of the world (locations of objects) with respect to the wedge human is currently focusing

[1] `https://aframe.io/`

on. The domain consists of three different actions:

$$A : \{left, right, confirm\}$$

where $left$ and $right$ actions are the guidance indicators, while the $confirm$ action checks if the object of interest is present in the currently focused wedge or not. Based on s, the policy generated from the RL approach (π) returns an action (a), which can be an AR guidance indicator to enable the human to find the object of interest in the remote environment. Once the agent performs the action, it observes the immediate reward (r) and the next state (s'). Using the tuple (s, a, r, s'), the agent updates the Q-value for the state-action pair. Finally, the agent updates the policy at runtime based on its interaction with the environment. We use the BURLAP library [2] for the implementation of our reinforcement learning agent and the domain.

Particle Filter for Human Intent Estimation: We use a particle filter-based state estimator to identify the motion intention of the human operator. In GHAL360, the particle filter is first initialized with M particles:

$$X = \{x^0, x^1, \cdots, x^{M-1}\}$$

where each particle represents a state, which in our case is one of the eight wedges. Each particle has an associated weight (ω^i) and is initialized as $\omega^i = \frac{1}{M}$. The belief

[2] `http://burlap.cs.brown.edu/`

state in the particle filter is represented as:

$$B(X_t) = P(X_t|e_{1:t})$$

where t is the time step, e is the evidence or the observation of the particle filter, and the belief state $B(X_t)$ is updated based on the evidence $e_1, \cdots, e_t$ obtained till time step t. In our implementation, the evidence of the particle filter is the current change in human head orientation (left or right), and the currently focused wedge. As the human starts to look around in the remote environment, based on the evidence the belief is updated:

$$B^{'}(X_{t+1}) = \sum_{x_t} P(X^{'}|x_t)B(x_t))$$

After every observation, once the belief is updated, we calculate the density of particles for all the possible states, and the state with more than a threshold percentage of particles (70% in our case) is considered as the predicted human intention. Once the particle filter outputs the predicted human intention (Line 26), we use a controller that outputs a control signal to drive the robot base toward the intended area based on the predicted human intention. In case the particle density does not reach the threshold in any wedge, the robot stays still until the particle filter can estimate the human motion intention.

6.2.3 Telepresence Interface

Based on the action suggested by the policy, GHAL360 overlays AR visual indicators over the current viewport. This results in a new frame $\mathcal{F}^{'}$ which is pre-

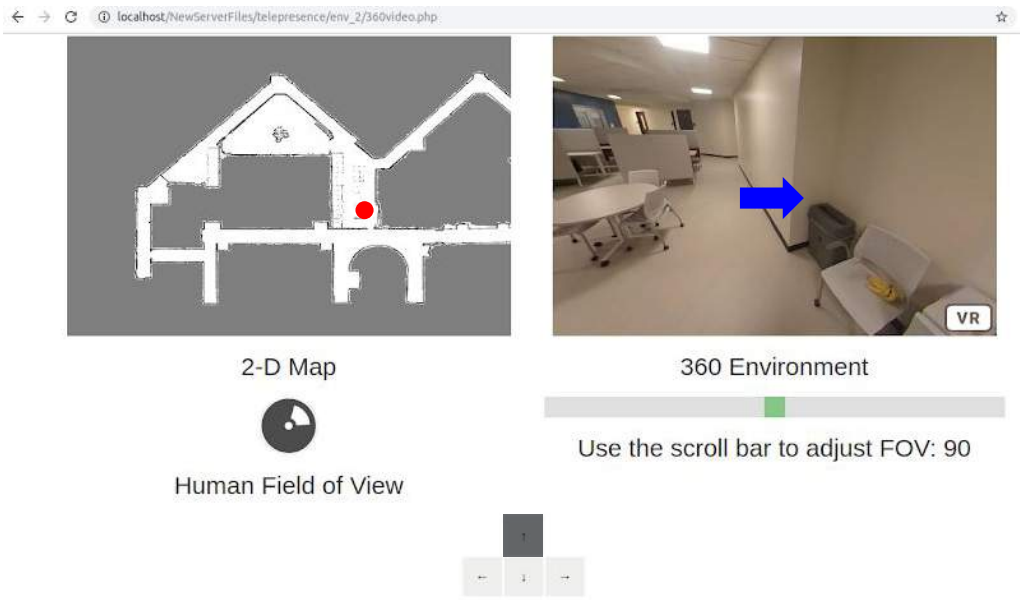


Figure 6.2: *Telepresence Interface* showing the 2D map, the frame of the remote environment, a 2D icon to show human head orientation, a scrollbar to adjust field of view, and a set of icons to indicate the current teleoperation commands given by the human operator.

sented to the user via the interface (Figure 6.2). The operator can use "click" and "drag" operations to look around in the remote environment. The telepresence interface provides a scroll bar to adjust the field of view of the human operator. Additionally, the users can easily change the web-based visualization to an immersive virtual reality 360° experience via head-mounted displays (HMDs) via a **VR** button at the bottom right of the interface. In the VR mode, the users can use their head motions to adjust the view of the remote environment.

The telepresence interface also shows the 2D map of the remote environment. Based on the robot's pose, a red circle is overlaid on the 2D map to indicate the live robot's location in the remote environment. Additionally, we show a 2D icon to convey the head orientation of the human operator relative to the map. The 2D icon also shows the part of the 360° frame which is in the field of view of the

human operator (Figure 6.2). The controller in Line 27 converts the estimated human intention into a control command. The control commands are converted to a ROS "teleop" message and are passed on to the robot to remotely actuate it. Based on the control command, the robot is remotely actuated and the 2D map is updated accordingly to show the new robot pose.

6.3 Experiments

We conducted experiments to evaluate GHAL360 in a target-search scenario where a human was assigned the task of finding a target object in a remote environment using an MTR. We designed two types of simulation environments, 1. *Abstract simulation*, and 2. *Realistic simulation*. To facilitate learning, we designed *Abstract simulation* to enable the agent to learn a goal-oriented policy for guiding human attention. The learned system is then evaluated in *Realistic simulation*. The main difference between *Abstract simulation* and *Realistic simulation* is that the *Abstract simulation* eliminates the real-time visualization to speed up the learning process.

6.3.1 Abstract Simulation

We use the abstract simulator to let the agent learn a strategy to guide human attention toward an object of interest in a 360° frame. In our abstract simulation domain, we simulate the process of GHAL360 interacting with the "world" as a Markov decision process (MDP) [10]. The transition probability of the human following GHAL360's guidance (*left* or *right* actions) is 0.8 in abstract simulation. There is a 0.2 probability that the human randomly looks around in the remote environment. The transition probability of the $left$ action and the $right$ action is

(a) (b) (c) (d) (e) (f)

Figure 6.3: (a)-(b), Home dataset from Matterport3D; (c)-(d), Office dataset from Matterport3D; and (e)-(f), Office dataset collected using a Segway-based mobile robot platform.

0.8 while the transition probability for the $confirm$ action is 1.0.

The reward function is in the form of $R(s, a) \rightarrow \mathbb{R}$. After the agent takes action $confirm$, if the value of W_0 (the wedge that the human is focusing on) is either w_2 or w_3, as defined in Section 6.2.1, the agent receives a big reward of 250. If the value of W_0 is either w_0 or w_4 after action $confirm$, the agent receives a big penalty, in the form of a reward of -250. Each indication action ($left$ or $right$) produces a small cost (negative reward) of 3 if the value of W_0 is w_0 or w_2. But if the $left$ or $right$ actions result in the value of W_0 becoming w_1 or w_3, then the actions produce a slightly higher cost (negative) of 15.

After executing the action (a), the agent moves to the next state (s') and receives a reward (r). The reward is based on the value of W_0 which can be one of the four values $\{w_0, w_1, w_2, w_3\}$ as described in Section 6.2.1, and W_0 is the wedge that the human operator is currently focusing on. If the agent takes action $confirm$ and transitions to s', where W_0 value is w_2 or w_3, the agent gets a reward as 250. But if the agent takes action $confirm$, and the value of W_0 is either w_0 or w_1, the agent gets a reward as -250. The actions $left$ and $right$ result in a reward of -3 if $W_0 = w_0$, whereas the reward is -15 if the value of $W_0 = w_1$ or w_3.

We trained the agent for 15000 episodes which resulted in the agent learning a policy for guiding human attention. We have evaluated the learned policy in the realistic simulation.

6.3.2 Realistic Simulation

We used the equirectangular frames from the Matterport3D datasets (Figure 6.3(a) - 6.3(d)) as well as frames from the 360° videos captured from the real

(a) Front (b) Back

Figure 6.4: Segway-based mobile robot platform (RMP-110) with a 360° camera (Ricoh Theta V), mounted on top of the robot, and laser ranger finder (SICK TiM571 2D) that has been used for prototyping and evaluation purposes in this research.

world (Figure 6.3(e) - 6.3(f)), and then simulated human and robot behaviors to build our realistic simulation. We used a Segway-based mobile robot platform to capture the 360° videos (Figure 6.4). The robot is equipped with an onboard 360° camera (Ricoh Theta V), and was teleoperated in a public space during the data collection process.

We model a virtual human in a realistic simulation that can also look around in the remote environment and send control signals to teleoperate the simulated robot using the interface. Moreover, as opposed to abstract simulation, the actions of the virtual human can be visualized in the interface, for example, if the virtual human looks to the left, the viewport is changed accordingly in the interface. The virtual human can both track the location of the simulated robot through the 2D map and perceive the 360° view of the remote environment via the telepresence interface. In our realistic simulation, the robot navigates in the 2D grid. If the virtual human tries to move to an inaccessible position, the robot stays still.

Based on the control signal, the location of the robot is changed, and the interface is updated accordingly.

6.3.3 Baselines vs. GHAL360

We compared GHAL360 using a target-search scenario with different baselines from the literature. We also conducted ablation studies to investigate how individual components of GHAL360 contribute to the system. The different systems used for comparison are as follows:

- **MFO**: Monocular Fixed Orientation based MTR, where the remote user needs to rotate the robot to look around in the remote environment [100].
- **ADV**: All-Degree View systems which are typical MTRs equipped with a 360° FOV camera. The human can use an interface to look around in the remote environment without having to move the robot base [110].
- **FGS**: Fixed Guidance Strategy system which consists of a scene analysis component along with a hand-coded guidance strategy that greedily guides the human using the shortest path to the target object.
- **RLGS**: Reinforcement Learning-based Guidance Strategy system uses the policy from RL to guide the human attention and suggests an optimal path for the human operator to locate the target object.

FGS and RLGS are ablations of approach, and the comparisons of FGS and RLGS with GHAL360 are ablation studies.

We used three different environments: two from the Matterport3D datasets, and one dataset collected from the real world. In every environment, there were

six sets of start positions for the robot in each of the environments, with each set including positions of the same distance to the target. For each environment, we selected two target objects, and each of the target objects was specifically selected to ensure that they were unique in the environment to avoid the confusion resulting from finding a different instance of the same object at different locations. For example, the target objects selected in environment 1 were "backpack" and "laptop", while the target objects selected in environment 2 were "bottle" and "television". The initial orientation of the robot was randomly selected, and a set of one thousand paired trials were carried out for every two-meter distance from the target object until 12 meters. The virtual human carried out one random action every two seconds out of "*move forward*", "*move backward*", "*look left*", and "*look right*" in the baselines (MFO and ADV). The actions "*move forward*", "*move backward*" are also carried out randomly in FGS and RLGS, while the actions "*look left*" and "*look right*" are carried out randomly only until there are no visual indications. In case, there are AR visual indications, the human follows the indications with a 0.95 probability. Using these actions, the simulated user could explore the remote environment. Once the target object is in the field of view of the human operator, the trial is terminated.

6.3.4 Illustrative Trial

Consider an illustrative trial where the human-MTR team needed to locate a laptop in a remote environment. Figure 6.5 shows the map of the remote environment with the robot's trajectory marked in blue color. The arrows in Figure 6.5 indicate the different human head orientations, and the red dotted lines point to the different viewports at these orientations. Figure 6.5 (a) shows the initial

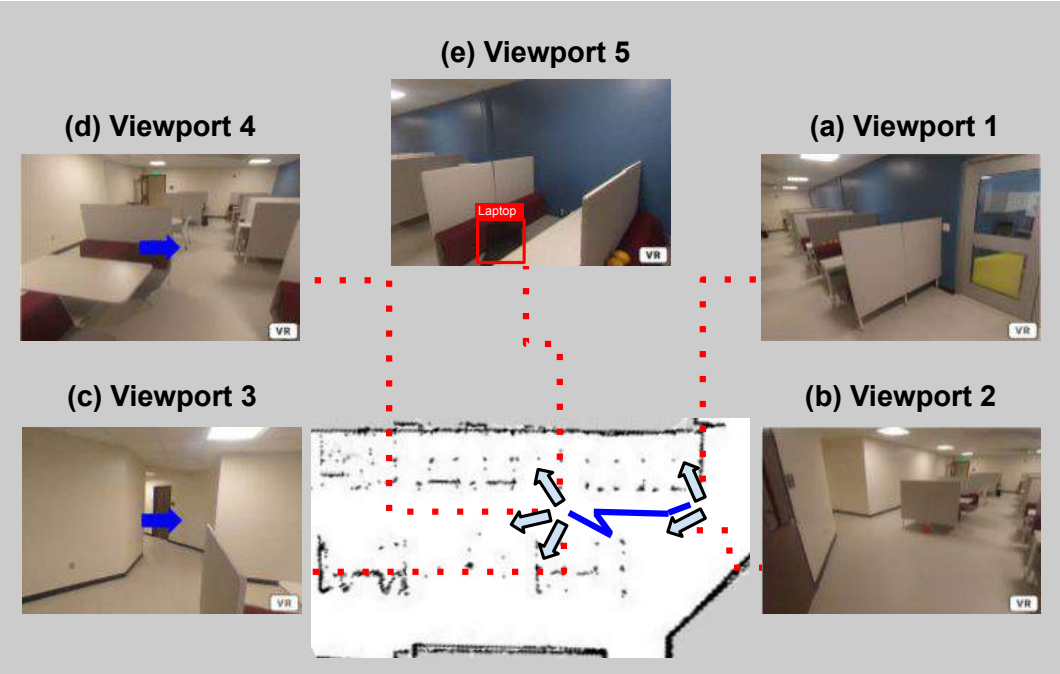


Figure 6.5: Map showing the trajectory of the robot in the remote environment looking for a "laptop." The arrows show the different human orientations along the trajectory; The images marked as (a)-(e) are the different viewports of the human at every orientation shown by the arrows.

viewport of a virtual human. The human started to look around at the remote environment to locate the laptop. Figure 6.5 (b) shows the second viewport after the human looked left. Then the human again looked right, and the viewport can be seen in Figure 6.5 (a). The human kept looking at the same position for the next two time steps. The particle filter estimated the human's intention, and accordingly, the controller output control signals to drive the robot to the next location.

The human continued to look around to find the laptop, and the particle filter constantly estimates the human motion intention resulting in the robot moving to different locations as shown by the blue marked trajectory in the map (Figure 6.5). Finally, the robot reached the location where the target object was located. As soon as the robot reached the location of the target object, the scene

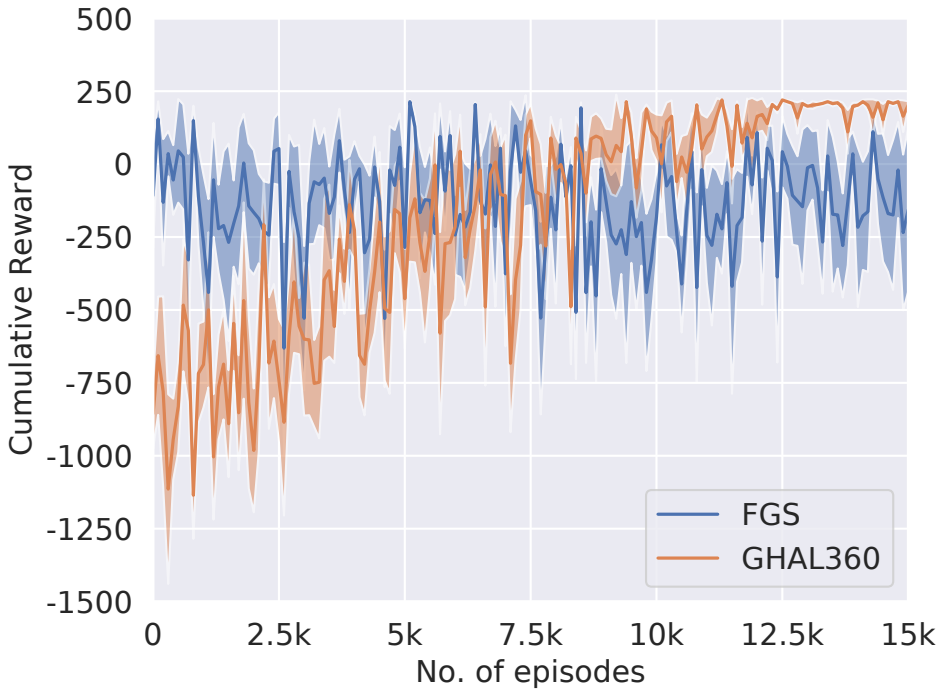
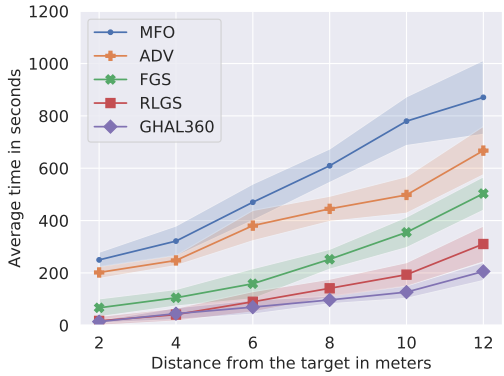
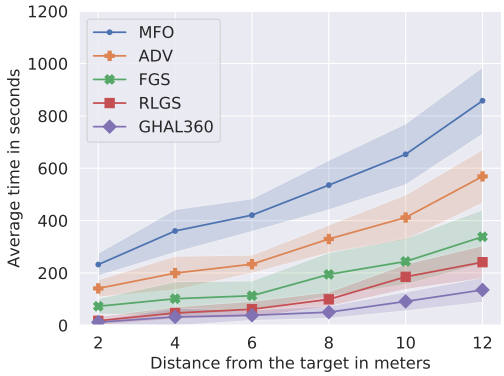


Figure 6.6: Comparison of the cumulative reward between FGS and GHAL360. Results show that the learned policy ultimately performed better than FGS (hand-coded baseline).

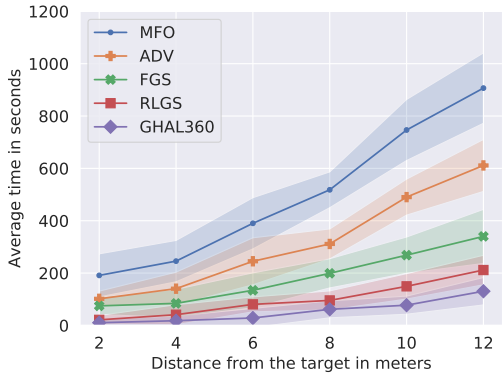
analysis component detected the laptop in the frame, and using the human head pose, GHAL360 overlayed the AR visual indicator over the viewport as seen in Figure 6.5 (d). The AR indicator suggested the human look right to find the laptop, but at first, the human did not follow the guidance and looked left. Again, the AR visual indicator was overlayed on the new viewport to guide the human attention (Figure 6.5 (c)). Now, at this time step, the human looked in the guided direction, and the viewport changed back to Figure 6.5 (d). Finally, the human followed the visual indicator and looked further right, and Figure 6.5 (e) shows the resulting viewport of the human with the laptop. Once the human found the target object, the trial was terminated.



(a) Env 1: Matterport3D (Home)



(b) Env 2: Matterport3D (Office)



(c) Env 3: Real Robot (Office)

Figure 6.7: Comparisons are made between GHAL360 and the other systems in terms of average time (y-axis) taken to find the target object from six different distances (x-axis).

6.3.5 Results

From the 15000 training episodes, we extracted 150 different policies after each batch of 100 episodes, and tested them in our realistic simulation. Figure 6.6 shows the average cumulative reward and standard deviations over five runs. We plotted the mean cumulative reward (Figure 6.6) of FGS and GHAL360 to compare the performance of hand-coded policy with the learned policy of GHAL360. The x-axis represents the episode number while the y-axis represents the cumulative reward. Figure 6.6 shows that in the early episodes when the agent is still exploring, the cumulative reward for GHAl360 is much lower than FGS. But with the increasing

number of episodes, it can be observed that GHAL360 outperforms FGS which represents the learned policy is superior to the hand-coded policy.

Table 6.1: This table shows the accuracy of different systems in target search tasks for different environments with different target objects. The bold font shows the best accuracy among all the systems, whereas the bold and italic font represents significant improvements.

Env (target)	Accuracy				
	MFO	ADV	FGS	RLGS	GHAL360
Env 1 (backpack)	.63 (.02)	.61 (.02)	.74 (.02)	.82 (.02)	**.84 (.02)**
Env 1 (laptop)	.65 (.01)	.66 (.02)	.75 (.02)	.82 (.02)	**.85 (.01)**
Env 2 (bottle)	.54 (.02)	.52 (.03)	.68 (.01)	.78 (.01)	**.81 (.02)**
Env 2 (television)	.58 (.01)	.59 (.01)	.71 (.01)	.80 (.03)	***.87 (.03)***
Env 3 (backpack)	.55 (.02)	.57 (.02)	.68 (.03)	.82 (.03)	.82 (.02)
Env 3 (laptop)	.56 (.02)	.56 (.01)	.69 (.02)	.80 (.01)	**.83 (.02)**
Overall	.58 (.04)	.58 (.05)	.71 (.03)	.81 (.02)	**.84 (.02)**

Figure 6.7 shows the overall performance of GHAL360 compared to the other systems enlisted above. Each data point represents the average task completion time (y-axis) for six different distances (x-axis) (i.e., randomly sampled positions in each of the six distances). One thousand trials were divided over five runs to plot each data point. The shaded regions denote the standard deviations over five runs for each of the six distances.

From the results, it can be observed that GHAL360 outperforms all the baselines (MFO and ADV) in terms of average task completion time at all six distances in finding the target object. Moreover, compared to FGS, GHAL360 performs better in all three environments and at six different distances denoting that the human guidance using the learned strategy outperforms the hand-coded greedy.

Furthermore, in addition to the learned guidance strategy, GHAL360 utilizes a particle filter to estimate human intention and guide the robot base toward the intended direction. In the bottom left of Figure 6.7(a) - 6.7(c), we can see that

the average time of task completion is similar between RLGS and GHAL360. This indicates that the particle filter in GHAL360 is less useful when the robot starts from a position that is close to the target objects. When the robot is distant from the target object, we see the particle filter produces a significant improvement, as evidenced by seeing the far right of the Figure 6.7(a) - 6.7(c). This observation confirms that the particle filter also contributes to the efficiency of GHAL360.

In addition to comparing the efficiency of different systems to GHAL360, we also compared the accuracy of target search tasks as shown in Table 6.1. It can be observed that the accuracy of GHAL360 is higher in finding all the target objects, except for the "backpack" object in Environment 3. Also, we observed significant improvement in accuracy in Environment 2 for finding the "television". This shows that along with the improvements in efficiency, GHAL360 also provides higher accuracy in finding target objects as compared to the other systems.

6.4 Summary

In this chapter, I describe our framework (GHAL360) [182, 183] that enables a mobile telepresence robot (MTR) to monitor the remote environment with all-degree scene analysis and actively guide human attention to the key areas at the same time. We conducted experiments to evaluate GHAL360 in a target search scenario. From the results, we observed significant improvements in the efficiency of the human-MTR team in comparison to different baselines from the literature.

7 Learning Visualization Policies from MARL

In this chapter, we aim to investigate more about learning AR visualization policies. In Chapter 5, we have leveraged imitation learning towards learning a visualization policy. While VARIL provides promising results, there are some limitations of VARIL, that we address in this chapter by enabling the AR agent to learn a visualization from reinforcements, i.e. interacting with the environment. This contribution falls under the proximate communication type, where human and robots are collocated.

7.1 Introduction

The persistent display of virtual objects can sometimes be extraneous for humans. Too many unwanted visualizations, or incorrectly placed virtual objects can lead to visual clutter. While AR, in general, enhances the human perception of the world by overlaying virtual objects onto the real world, such unnecessary augmentations can be distracting for humans while working alongside robots and can lead to degradation in task performance [77]. Moreover, AR interfaces can alter attentional focus and result in *inattentional blindness* [156]. Inattentional blindness is when humans tend to miss highly notable events or objects in their field of view which are noticeable otherwise [184]. There can be instances when the human is busy with his/her own tasks and such overhead of visualizations can

cause the human to lose focus, and in the long run discourage humans from using such distracting interfaces.

As discussed in the last chapter, visualization strategy can be considered as a policy that maps the current state to a visualization action that the visualization agent needs to take to minimize distraction and maximize the information delivered to the human. While designing a visualization strategy, there are a number of factors to consider, for example, which virtual objects to augment (do some objects have priority over others), how to augment the virtual objects (arrangement of the virtual objects), when to augment the virtual objects (human is available or busy), etc. In Chapter 5, we presented the first work that enables the visualization agent to learn a meaningful policy from expert demonstrations.

The imitation learning approach of VARIL provided several benefits including low exploration cost because the agent tries to closely mimic the expert demonstrations without having to carry out trial and error. Another benefit of mimicking expert demonstrations is that as the policy is trained on more demonstrations, the policy becomes more predictable. More importantly, since the expert provides feedback at runtime to the visualization agent, it can potentially correct the wrong behaviors quickly leading to a speedup in the learning process.

Although VARIL provided a strong base to support our direction toward learning visualization strategies for human-multi-robot teams, there exist some limitations in the framework that motivates our research to investigate different learning architectures. VARIL has several key limitations as follows, 1, the visualization agents' policy cannot account for unforeseen scenarios due to the lack of diversity of data provided by the expert; 2, the expert demonstrator needs to carefully

provide feedback to avoid the AR agent learning from incorrect feedback, but this increases the overhead tremendously on the expert demonstrator in a complex human-multi-robot system; 3, the data collection is too costly limiting the scalability of the system to more complex scenarios.

To overcome these limitations, we design a framework, called, ***Visualizations for Augmented Reality using Multi-agent Learning*** **(VARMAL)**, that enables human-multi-robot collaboration in a shared environment. The AR interface of VARMAL leverages AR visualization policies to enable humans to collaborate with a team of robots. The AR visualization policies of VARMAL are learned via multi-agent reinforcement learning, specifically, the QMIX algorithm [159], a novel value-based method that can train decentralized policies in a centralized end-to-end fashion. The AR agent interacts with the environment toward learning a policy that enables the AR visualization agent to suggest an action based on the local observations of each robot.

The visualizations of VARMAL were learned in the simulated warehouse with eight robots working alongside humans in a human-robot collaborative delivery task. We have evaluated VARMAL using where a team of robots works to collaborate with a human for delivery tasks, and the tasks for both humans and robots are non-transferable. The robots are tasked with delivering objects to different locations (drop stations) and the human helps unload the objects from the robots waiting at drop stations. The learned AR visualization policies were compared with the static AR policy for human multi-robot teams from the literature [163], and a handcrafted policy, and the results suggest that the VARMAL policies significantly decrease the robot wait time. Results indicate that the learned visualiza-

tion policies of VARMAL significantly increase the efficiency of the human-robot team in task completion.

The next section provides details on the experiments conducted to evaluate VARMAL in the human-multi-robot collaboration scenario.

7.2 Experiments

We conducted experiments in a simulated warehouse environment that we discussed in Chapter 8. We have compared VARMAL with several baselines from the literature to evaluate the hypotheses that *VARMAL increases the collaboration efficiency in human-multi-robot teams.*

7.2.1 Experiment Setup

We designed the experiments in a simulated warehouse environment where the human participant had to collaborate with a team of eight robots to complete delivery tasks. The robots were given the task of picking boxes from the shelves and dropping them at assigned drop stations in the warehouse environment. Each robot is assigned three boxes for each trial and once the last box has been dropped off by the robot, it will begin navigating to its starting position. The robots wait at the drop station with the boxes to wait for help from the virtual human. The virtual human is tasked to navigate to one of the drop stations to help unload the box from the robots waiting at the drop station. This delivery task requires human-robot collaboration because the robots cannot drop the boxes with the help of the virtual human. The trial is completed once all eight robots have delivered their three boxes and have reached their starting position.

Policy Learning: VARMAL learns the visualization policy by utilizing QMIX [159]

in the simulated warehouse environment. In the simulator, we model the interactions of agents with the world as a POMDP [11] to account for the noise in the environment.

7.2.2 Formalization of VARMAL

In this subsection, we describe the state space, action space, reward function, and the network architecture for the AR agent of VARMAL for learning a visualization policy.

In VARMAL, the observations of each robot have the following features:

- humanState: {close, moderate, far}
- robotTaskState: {picking, dropping, none}
- robotRemainingTasks: {few, many, none}
- robotWaitingTime: {short, medium, long, none}
- nearbyRobotVizStatus: {few, many}

Every agent has a limited range, which can also be termed as the agent's field of view (FOV. The agent can observe other agents in its FOV but since the range is limited, the environment is partially observable to each agent. The above state representation represents a rich context for the entire human-multi-robot system from each agent's perspective.

In our implementation, the range of each robot is set to a radius of five meters. For *humanState*, if the human is within the FOV, we characterize it as *close*, for distances less than eight meters - *moderate*, and more than that is considered as *far*. The *robotTaskState* tracks if the robot is *picking*, or *dropping*, or in case of

task complete, the observation is *none*. In the case of *robotRemainingTasks*, *few* denotes [0,1] tasks, while *many* represents more than one remaining delivery task, and the value is set to *none* when the robot has completed all the delivery tasks. To represent the robot wait time at a drop station, we use the following metrics: $\{short < 20;\ 20 < medium > 35;\ long > 35\}$, where the time is in seconds. For each agent, *nearbyRobotVizStatus* categorizes the number of robots in its vicinity that have their visualizations turned on. The values are as follows: $\{few <= 2;\ many > 2\}$.

In our domain, every robot needs to take decisions for three types of visualization actions (action space) which can be either turned on or off, that form the action space:

- liveLocationAvatar: {on, off}
- trajectory: {on, off}
- transparentAvatar: {on, off}

In terms of the visualizations of the balloon, the state-space of our AR agent for each drop station consists of the following:

- humanState: {close, moderate, far}
- robotsWaitingTimeDS: {short, medium, long}
- nRobotsAtDropStation: {few, many}

Here, the AR agent observes the wait time of robots at its drop station (*robotsWaitingTimeDS*), where the values are: $\{short < 30;\ 30 < medium < 40;$

long $> 40\}$. The AR agent also tracks the number of robots at the drop station (*nRobotsAtDropStation*), where $\{few <= 2; many > 2\}$.

Finally, for each drop station, the AR agent can enable or disable the balloon on the drop station, hence the action space is represented as follows:

- balloon: {on, off}

In addition to the above state space, the QMIX architecture also consists of the global state that is available during the centralized training phase. The global state consists of the *lastAction* taken by each agent, which is a tuple.

The goal of the learned visualization policy is to minimize the robots' wait times while at the same time delivering the maximum information with the least amount of visual clutter. We use the following rewards in our environment, where,

- liveLocationAvatar: {Living: -1, Start: -5, End: -5}
- transparentAvatar: {Living: -2, Start: -5, End: -5}
- trajectory: {Living: -2, Start: -10, End: -10}
- balloon: {Living: -2, Start: -20, End: -20}

The above reward structure shows the living cost, and the cost to start and end the visualization actions. In addition to the above reward structure, the robot agents get a reward of 100 for dropping the object, and a penalty of -1 for waiting at a drop station. The AR agent for the balloon receives a reward of 100 for every robot that completes dropping a box at the drop station, and a penalty of -1 for each robot waiting every second at the drop station.

Using the above state and action space, the VARMAL framework learns a visualization policy for human-multi-robot teams. Depending on the implementation

of the different visualizations for the AR agent, the state space and action space can vary to incorporate these visualizations.

Network Architecture: The agent network architecture of VARMAL for each agent network is a DRQN. Similar to the default implementation of QMIX [159], we use a recurrent layer comprised of a GRU with a 64-dimensional hidden state, with a fully-connected layer before and after. The value of ϵ is annealed linearly from 1.0 to 0.05 over 16000 time steps, and then the value stays constant during the learning process. The value of γ is kept constant at 0.99 Our replay buffer stores the last 600 episodes, and we sample batches of 32 episodes uniformly. We update the target networks after every 60 training episodes. Every episode is set to a maximum length of 60-time steps, and if the episode exceeds the time steps, we reset the environment because in most cases the robots or the human get stuck in an obstacle in such episodes. Our mixing network also follows the default QMIX architecture, with a single hidden layer of 32 units using an ELU non-linearity. A single hidden layer of 32 units with a ReLU non-linearity is used for the hypernetwork that produces the final bias of the mixing network.

Baselines: We compared VARMAL with four baselines to evaluate the effectiveness of the learned visualizations. The different systems used for comparison are as follows:

- **NoViz**: This system does not provide the virtual human with the robot status information, so we call it as No Visualization (NoViz).
- **ARROCH**: ARROCH enables the virtual human to track the status of all the robots by keeping all the visualizations on all the time (Chapter 4).

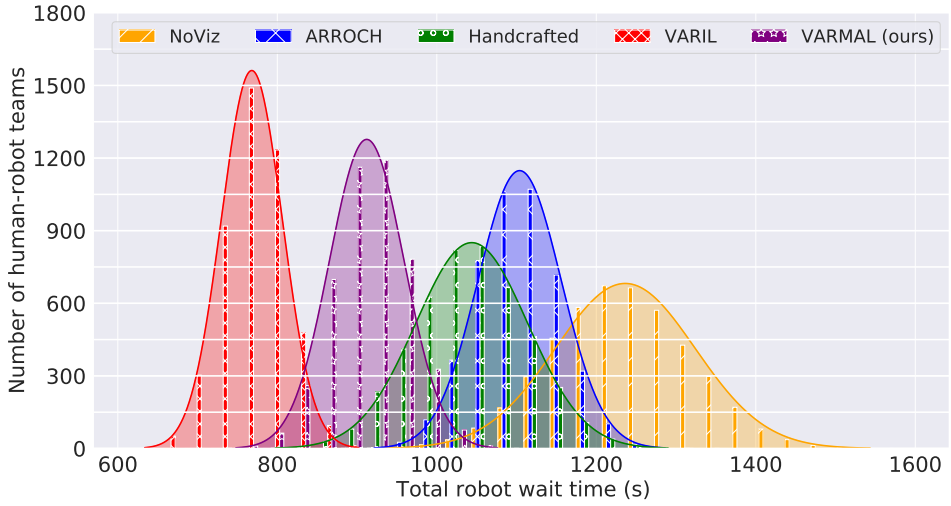


Figure 7.1: Task completion time

- **Handcrafted**: This system deploys a handcrafted visualization policy to help the virtual human track the robots.

- **VARIL**: This system deploys a learned visualization policy using expert demonstrations (Chapter 5).

7.2.3 Results

Objective Results: Figure 7.1 shows the histogram, where the x-axis represents the total wait time of all the robots in seconds, and the y-axis represents the number of human-robot teams with the corresponding robot wait times. From the figure, it can be observed that for most human-robot teams, VARMAL had the shortest robot wait times as compared to NoViz, Handcrafted, and ARROCH. The figure also shows the relative comparison between VARMAL, and VARIL, where VARIL performs better than VARMAL, which implies that the policy generated from human expert demonstrations performs slightly better than the policy generated by VARMAL. Additionally, we plotted the Gaussian curves that show the

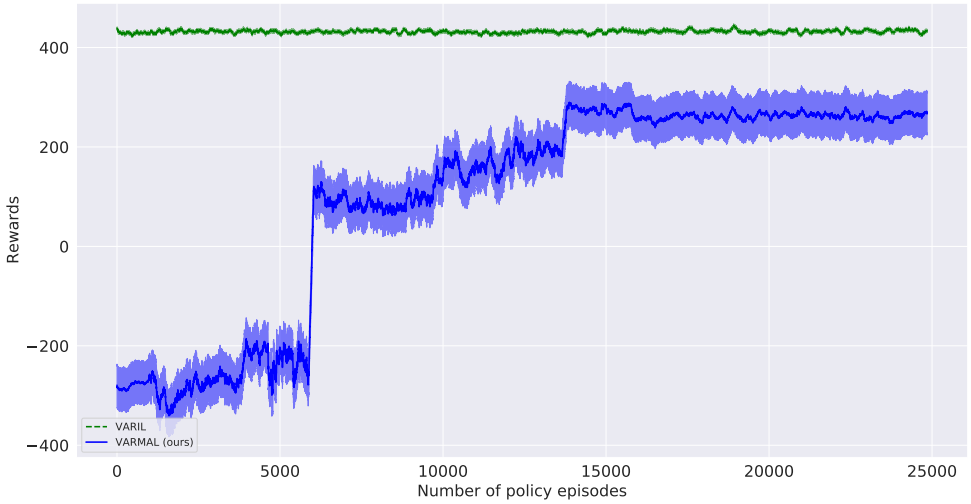


Figure 7.2: Learning curve

distribution of the data points for all the methods. We also analyzed the statistical significance, in every trial, we sum up the task completion time of all the robots. We found that VARMAL performed **significantly better** than all the baselines except VARIL, where $0.01 < p - value < 0.05$. Finally, in comparison to the total robot wait times, the wait times for robots in VARMAL were significantly shorter than the baselines. All of these results support Hypothesis-I which states VARMAL improves the human-robot team's task completion efficiency.

Policy Evaluation: We have compared VARMAL with the visualization policies of VARIL. Figure 7.2 shows the number of episodes on the x-axis and the cumulative reward on the y-axis. From the figure, it can be observed that VARMAL's early policies had negative rewards. This is because the agent incurred the living cost of the visualizations, as well as the start and end costs frequently. As the policy of VARMAL evolves, the agent learned to balance the visualization frequency leading to higher rewards. Furthermore, the figure shows that VARMAL's reward grew closer to VARIL's reward and reached a peak at about 18000 episodes. Even

though the rewards for VARIL were higher, it was expected because we deployed the final policy of VARIL learned from human expert demonstrations. The figure indicates that VARMAL was successful in learning a meaningful policy without the human demonstration dataset, achieving near-optimal performance in learning visualization policies for human-multi-robot teams.

7.3 Summary

In this chapter, I present our framework, VARMAL (*Visualizations for Augmented Reality using Multi-agent Learning*), that for the first time introduces a MARL-based AR visualization strategy for human-multi-robot collaboration. The framework learns a decentralized visualization policy which is then deployed in a centralized manner, where the AR agent dynamically selects a visualization action based on its local observation. We have compared VARMAL with various baselines from the literature and the results suggest that VARMAL increased the collaboration efficiency of human-multi-robot teams in team task completion time. Moreover, additional studies with comparison to the policy learned from human expert demonstrations, suggest that the AR agent of VARMAL learned a near-optimal policy from reinforcements.

8 Benchmark Simulation Platform for AR-mediated HRC

In this chapter, I present the open-source benchmark simulation platform that for the first time enables the simulation of the human-multi-robot collaboration and their interactions mediated by AR. The simulator consists of many robotic platforms and virtual human models with task and motion planning capabilities, a few virtual AR devices, and many interactive objects. This contribution tackles the proximate communication type, where humans and robots are collocated, and we provide three different environments each with unique human-robot collaborative tasks.

8.1 Introduction

Robots are increasingly ubiquitous in our everyday environments including offices, warehouses, hotels, as well as homes. The advancements in robot autonomy have enabled the robots to execute long-horizon tasks without the need for human supervision. While the robots can accomplish a wide variety of tasks autonomously, there are still scenarios where the robots need human help. For humans and robots to collaborate, they need to efficiently convey their state, as well as their intentions. Human–robot Interaction (HRI) is a field of study dedicated to understanding, designing, and evaluating robotic systems for use by or with

humans [1]. Researchers have designed different modalities of communication to enable robot and human teammates to exchange information.

Augmented Reality (AR) has provided a communication medium to convey critical spatial information between humans and robots [163]. Our world is by default 3D in nature, so leveraging spatial information can enable human-robot teams to better correlate the shared information to their environment. The growing attention to Augmented Reality/Virtual Reality based HRI can be witnessed with the emergence of the VAMHRI community in recent years [185]. Researchers have designed different AR-based systems to enable humans to understand the internal state of the robots [70]. AR has also been utilized to convey the robot's motion intent to the human counterpart [65, 186, 187]. Recently, AR has been leveraged in human-multi-robot teams towards increasing the efficiency of human-robot collaboration [163].

Several challenges arise when evaluating human-robot systems in the real world, which include safety implications, the additional burden of managing robotic hardware, costs, etc. These challenges aggravate when working with human-multi-robot systems, which can impede the appropriate evaluation, and in turn the deployment of these systems in the real world. To alleviate these issues, researchers have developed several simulation platforms that can facilitate the extensive evaluation of human-robot systems. When integrating AR in the human-robot systems loop, the complexity of the system increases, and AR introduces new challenges like user motion sickness, user fatigue, etc. Simulating AR can enable quick prototyping of the interfaces while reducing the need to deploy low-ranked interfaces from the evaluation conducted in simulation to attenuate the challenges associated

with AR. One of the major advantages of simulating AR is to enable learning in While the existing simulators support the simulation of human-robot teams, they all lack the AR component which motivates our research in this direction.

Very recently, researchers have developed simulators that try to integrate AR visualizations in simulated environments [75, 139, 137]. But all of these existing simulators do not provide an actual simulation of AR visualizations, where the visualizations only work when the AR objects are in the field of view, which limits the capabilities of using actual AR visualizations. To address the limitations of the existing simulators, we develop, *Multi-Agent Augmented Reality Simulator (**MAARS**)*, which enables the simulation of the AR agent that provides a communication medium within human-multi-robot teams. MAARS will enable fast prototyping of AR visualization techniques, and enable extensive evaluation to facilitate the quick implementation of robust AR interfaces. Moreover, MAARS will serve as a benchmark for the growing community of AR and HRI researchers, and accelerate the research towards bridging the gap between research conducted in the lab, and the deployment of these systems in the real world.

The benchmark platform consists of three different environments (shopping mall, office, and warehouse), each with a pre-designed human-robot collaboration task. Additionally, the platform consists of different robotic platforms for the robot agents (turtlebot, four-legged spot robot, fetch robot, and segway-based mobile manipulator) with task and motion planning capabilities. MAARS also provides several models for virtual human agents with perception capabilities along with task and motion planning capabilities. Finally, MAARS consists of different types of AR agents (2D screen-based devices like tablets, and AR headsets), as well

as the implementations of several AR interfaces from the literature. All the pre-designed tasks in the environments support quantitative evaluation by storing the task-related metrics.

8.2 MAARS Architecture

MAARS enables the simulation of multi-robot human collaboration behaviors mediated by Augmented Reality. In this section, we describe how we simulate the three different agents of the MAARS simulator:

- Multi-robot teams
- AR devices
- Virtual humans

We also describe the interaction of these agents with each other, and with the simulated environment.

8.2.1 Simulation of Multi-Robot Systems

MAARS can simulate multi-robot teams that collaborate with other, with the environment, and also with the other agents in the environment. Every robot has its own task planner, and a motion planner that enables it to execute tasks in the environment. MAARS supports different robotic platforms including Turtlebot2, Fetch robot, four-legged SPOT robot, and a segway-based mobile manipulator (Figure 8.2.1).

Motion Planner: The motion planner in MAARS is based on the Unity NavMesh (short for Navigation Mesh) [165] which enables the robots to navigate while avoiding

Figure 8.1: Different robotic platforms available in the MAARS simulator.

obstacles as well as other agents. A NavMesh is a data structure to describe the navigable surfaces of the environment and allows the robot to find a path from one navigable location to another in the environment. For every environment, MAARS provides a pre-generated NavMesh, as well as an API to generate NavMesh at runtime because the environments are configurable, and need an updated NavMesh when the default parameters of the environment are modified.

Task Planner: We have a centralized task planner, which allocates tasks from a set of remaining tasks in the task pool. The robots constantly share their task information with their human counterparts while working on their tasks. The human counterpart can provide feedback on the tasks, and based on the feedback, the robots can adapt to the feedback by triggering replanning. Such replanning enables the robots to generate a plan at runtime that addresses the feedback of the human counterpart.

8.2.2 Simulation of AR devices

One of the key contributions of the MAARS simulator is the simulation of the AR agent. MAARS simulates two different kinds of AR agents, 1, *AR tablet*, and 2, *AR glasses*, to address different use cases. In the real world, AR agents use a camera to track the actual world, and overlay augmented objects at specific locations to provide a composite view to the humans. Similar to real AR devices, we use a simulated camera that captures the simulated world and displays it using the AR agents' screens. While the AR tablet agent consists of a large screen, where the human can see the live feed captured using the camera, the AR glasses agents, are designed to mimic the FOV of view of human eyes, and display the live camera feed similar to human FOV.

The AR agents have a 6 degree of freedom to enable the human to move these AR agents around in the 3D space. The reason behind creating two different types of AR agents is the availability of two different classes of AR devices available in the real world. We create the AR tablet agent to mimic the mobile AR devices, while the glasses fall into the class of wearable AR devices. On the one hand, AR mobile devices require the human to occupy one/both of their hands while using, they can be used in a passive manner, where the AR device can rest on a surface, while on the other hand, the wearable devices provide the flexibility of hands-free usage at the cost of comfort since wearing such devices for long periods of times has a number of issues.

The simulated AR agents act as a mediator between the team of robots and the humans. On one hand, the AR agents help the human to track the status of the team of robots, while on the other hand, the feedback from the human operator

is communicated back to the team of robots to enable them to replan based on the human feedback. The bidirectional communication of AR devices is facilitated by the different APIs created in MAARS. Currently, the AR agents in MAARS receive the following input from each robot, 1, Live location, Current plan, and 3, Robot type. Based on this input, the AR agents augment different 3D objects to enable the human operator to visualize the status of the robots, and their future intentions. From the human side, the AR agents, take the human feedback if available, and relay them back to the team of robots. In the next subsections, we describe the different 3D visualizations available in MAARS to visualize robots' status, as well as the status of the environment.

Different visualizations seen in AR agents: Once the Unity receives the live locations and robot plans, they are overlayed onto the simulation environment. We use 3D robot models to show the live locations, while we use trajectory markers to show the plans of the robots in Unity. Since both the Unity and Gazebo environments are identical, the locations and plans of the robots from Gazebo are easily overlayed in Unity.

To visualize the robots' current states (live locations) and intentions (motion plans), we design two types of AR devices, a tablet, and an AR headset. Similar to a real-world AR device, the AR devices in Unity, also have a screen that can be used by the virtual human to visualize the AR content. Also, the AR devices have a real camera to capture the simulated environment and then place the AR content over the camera feed. The field of view (FOV) of the AR devices can be tuned, but as per the realistic devices, the FOV of the AR tablet's camera is set to 90 degrees, while the FOV for the AR headset's camera is set to 110 degrees.

8.2.3 Why simulating AR is not straightforward

Several robotic simulators that provide mixed-reality functionality do not enable the visualization of virtual objects that are occluded. In the real world, humans can visualize virtual objects (robot plans or locations) even through obstacles because the virtual objects are overlayed over a live stream of the real world. This enables humans to see-through walls or other obstacles which helps to track the status of the robots in complex multi-room environments too. One straightforward approach could be to make the walls transparent to provide the functionality of making the virtual objects visible through walls, but this fails if the virtual object is behind anything other than the wall. Moreover, increasing the transparency of any object in the simulated environment will also lead to a mismatch in the actual visualization of the virtual objects due to a layer of transparent objects on top of the virtual objects. Finally, the biggest problem in making the walls transparent is visual clutter because the human will see all the 3D objects behind the walls, and distinguishing AR objects amongst them is very difficult.

Our approach to providing AR functionality in the simulator mimics the nature of AR devices in the real world. All the virtual objects are overlayed on top of the simulated environment, enabling the virtual human to see-through obstacles and track the robots.

8.2.4 Simulation of Virtual humans

To interact with the robots, we model the third type of agent in MAARS which are the virtual humans that can perceive as well as actuate in the environment.

Figure 8.2: Different virtual human avatars available in the MAARS simulator.

The human agent leverages a similar task and motion planner as the multi-robot agent teams. While the human agent has its' own set of tasks, we also design scenarios where the human agent collaborates with a team of robots to complete a common task. The human agent can also move around in the environment while utilizing the AR agents' capabilities. MAARS showcases the use of both the AR tablet agent (Figure) as well as the use of AR glasses agent (Figure) in the bundled environments. The human agent in MAARS is designed using 3D humanoid avatars providing a range of joints that can be actuated, for example, the legs for moving in the environment, and the hands to interact with the objects in the environment. Similar to the AR agent, the human agent also consists of a camera that mimics the functions of the eyes. Moreover, we use a 110-degree FOV for the human agent's camera which is similar to those of the real-human FOV. The human agent can move its head to look around in the environment using this camera. The camera enables the human agent to perceive the environment and visualize the information shared using the AR agents. The human agent is able to carry a soda can, open a door, or even perform a box handover with the robots.

Another key contribution of MAARS is how the human agent interacts with the

environment by utilizing the AR agents' capabilities. Firstly, MAARS constantly tracks the human agent's head to know where the virtual human is currently looking in the environment. MAARS also tracks the focus of the human agent to narrow down the region of interest in the wide FOV of the human agent's camera. In the next subsection, we describe how we model the perception in the human agent.

Perception of human agent: MAARS human agents can visualize the environment directly with their camera or visualize the environment through the AR agent's visualizations. Since AR visualizations are overlayed over the environment, it is vital that the human agent can differentiate between the objects observed directly and via the AR agent. Such differentiation can enable the researchers to understand if the 3D object is occluded by another object, and how the human agent's behavior is affected with and without the use of visualizations provided by the AR agent. For this, we design a simple algorithm that enables MAARS to differentiate if the human is looking at a real or an augmented occluded 3D object.

The algorithm initially tracks all the objects in the FOV of the human agent. Then, for every object in the FOV, we calculate the actual distance between the human agent's camera and the 3D object. Once, the distance is calculated, a laser beam is used to point from the human agent's camera towards the current 3D object. The laser beams in Unity return the distance where the first 3D object was hit in the path of the beam. Finally, we compare the actual distance and the distance where the laser beam was hit, and if the distance is less than the actual distance, the object is occluded by another object.

In the next subsection, we briefly delineate how researchers can leverage AR information to influence the behavior of the human agent.

Human agent behavior modeling: MAARS enables the human agent to utilize AR information to decide how it interacts with the environment. Consider a simple scenario where there are two workstations in the environment that can be used by either the human agent or a robot agent at a given time. If the human agent can track where the robot agent is heading using the AR visualizations, the human agent can avoid running into the robot agent, and effectively decide which workstation it should visit. AR visualizations can also be used to build synergy where a human agent can observe the AR information to track the robot agent, and decide the next location so that it can navigate to the robot's location. The human agent in MAARS can also provide feedback using the AR agent. Again, such behaviors can enable the team of robot agents to take the human agent's feedback into consideration and may lead to synergies.

Now, in the next subsection, we describe the three different environments that we built to enable human-multi-robot collaboration behaviors.

8.2.5 Simulation of the Environments

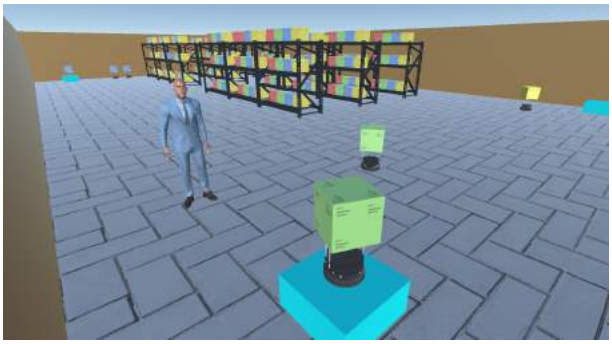
We have designed three different environments to diversify the use cases of the simulator including an ***office environment***, a ***warehouse environment***, and a ***shopping environment***. Figure 8.3 shows all three different environments including the three agents of the MAARS simulator. Now, we will describe the different tasks designed for each environment to enable human multi-robot collaboration behaviors.

Office environment: We created an office environment, where a virtual human worked on its own task of solving a Jigsaw puzzle (designed to imitate an assembly task). On the other hand, a team of robots worked on a delivery task, where the robots had to pick up objects from designated locations, and drop them at a drop station (a room where the virtual human was working on its tasks). We designed a human-robot collaboration scenario, where the virtual human helps the robots by opening the door, and letting the robots enter the drop station. The role of AR here is to enable the virtual human to track the status of the robot, and help the robots when needed. Since the virtual human is also doing their own task, the information obtained using AR enables the human to efficiently decide when to open the door to avoid affecting its' own task time.

Warehouse environment: One of the other environments simulated in MAARS is a warehouse environment, where a team of robots works alongside humans. The warehouse consists of a number of shelves, and each shelf consists of several boxes. The warehouse also consists of a number of drop stations, where the boxes need to be dropped. The warehouse environment consists of different parameters that allow scaling the number of shelves, the number of robots, the type of robots, as well as the number of drop stations. Such scaling can enable the researchers to train policies on different sizes of warehouse, while it also provides a way to replicate the size of a real warehouse to facilitate sim-to-real transfer. The robots' task in the warehouse is to pick up the assigned boxes from the shelves, and drop them at the designated drop stations. The task of the virtual human is to help the robots unload the boxes at the drop stations. Since there are a number of drop stations where the robots might possibly wait for the virtual human for unloading,

(a) Environment 1: Shopping Mall

(b) Environment 2: Office

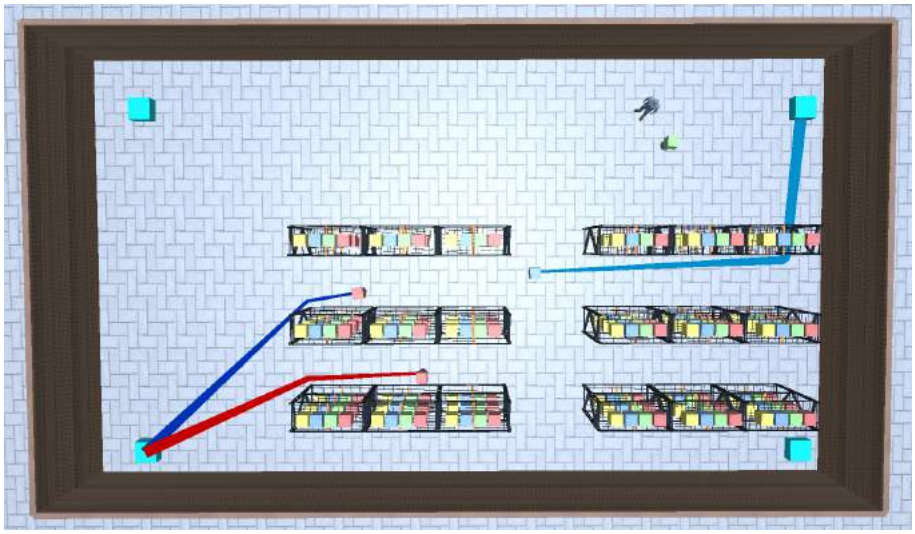


(c) Environment 3: Warehouse

Figure 8.3: Different pre-designed environments of the MAARS simulator.

the virtual human tracks the robots using AR device to know the locations of the robots. Using such information, the virtual human can efficiently decide which drop station requires its attention, leading to a decrease in robot wait time, while increasing the efficiency of human-robot teams.

Shopping environment: Another use case that we designed, is a shopping mall environment where the virtual human can order from different vendors. We designed realistic outlets for vendors that contain a number of interactable objects. An example of interactable objects can be seen in Figure 8.3 where a robot is car-



(a) Top-view of the simulated warehouse environment

(b) First-person view of the virtual human tracking robots in AR

Figure 8.4: Figure shows the top view and the first-person view of the virtual human for the warehouse environment, where the virtual human can leverage AR to track the status of robots.

rying a can of soda to a virtual human. We designed different AR visualizations that assist the virtual human to order from different vendors. The virtual human can look around to see the floating names of different vendors, their products, and the estimated time of delivery for them. Once, the order is placed, the virtual human can track the location of the delivery robot, the path taken by the robot, as well as the estimated time it will take for the delivery. We designed this environment to showcase how AR can help us to complete the simplest of tasks such

as ordering. We believe such environments can foster researchers to experiment with different visualizations in shopping use cases.

Example of MAARS simulator: We provide a short example of the functioning of the MAARS simulator, where a human collaborates with a team of robots in the warehouse environment. The virtual human uses a simulated AR headset to track the status of the robots. The task for the human-robot team is the same as described in the warehouse environment details. Figure 8.4(a) shows the top-view of the environment with the team of robots working on delivery tasks, and their AR visualizations. A virtual human tracks the status of the robots, and helps them unload boxes at the drop stations. The first-person view of the virtual human can be seen in Figure 8.4(b).

8.3 Summary

In this chapter, I present our open-source simulator that enables the simulation of human-multi-robot teams working on collaborative tasks, and their interactions mediated by AR. Our simulator provides many robotic platforms and virtual human models with task and motion capabilities, a few virtual AR devices, and many interactive objects. Our novel simulator could serve as a benchmark platform for the AR-HRI community to advance the research without the need to explicitly run experiments on real hardware all the time. Moreover, the benchmark platform consists of tasks that can serve as the base for the research community to compare their research to the state-of-the-art AR-based algorithms and interfaces in a common setting.

9 Conclusion

In this chapter, I conclude the dissertation and provide a summary of the contributions toward bridging the observability gap in human-robot teams by leveraging AR visualization policies.

A novel framework, called, ARROCH, was designed to enable bi-directional human-multi-robot collaboration. Additionally, the framework enables the human teammate to give feedback on the robots' plans. Finally, the feedback is utilized toward replanning leading to better collaboration efficiency. The AR interface of ARROCH improves the observability of the human over the team of robots by providing the robots' current and future task-related information. On the other hand, the robots' observability over the human's task is improved by utilizing the feedback from the human.

ARROCH employed a static visualization policy, that constantly provided all the visualization for a team of robots. With the increase in the number of robots, the visualizations can be overwhelming for the human teammate to track the status of the robots. Toward overcoming these limitations, we designed a framework, called VARIL, that learns a visualization policy for human-multi-robot teams. The visualization policy was learned using imitation learning, where a human expert provided demonstrations to train the visualization policy. VARIL enhances ARROCH visualizations, and bridges the observability gap by reducing the dis-

traction level of human users, and increasing the efficiency of human-robot teams.

While VARIL was the first work in the literature to learn AR visualization policies, several issues related to imitation learning entail VARIL. To address these issues, I designed a framework called, VARMAL, that leverages multi-agent reinforcement learning to learn AR visualization policies. Since we have a multi-agent environment, agents can leverage their skills to collaborate to learn visualization policies toward elevated human-robot collaboration. The visualization agent of VARMAL, interacts with the environment and other agents, and learns a policy by reinforcement resulting in the elimination of the need for expert demonstrations which are needed in VARIL. VARMAL was compared with several baselines from the literature, and the results suggest that VARMAL enables efficient human-robot collaboration, while also decreasing the visual clutter.

Apart from bridging the observability gap for human-robot teams in a shared environment, I designed another framework, called, GHAL360, that leveraged AR visualizations to elevate human-robot collaboration in shared embodiment scenarios. GHAL360 enabled the human operator to remotely teleoperate a robot, while enhancing the human operator's situational awareness of the remote environment. We leveraged reinforcement learning to learn a visualization policy that guides human attention to areas of interest outside the FOV of the human, bringing the observability of the human operator, a step closer to robot's observabilities.

The contributions of this dissertation have a shared goal of bridging the observability gap in human-robot teams, where each of the contributions tries to bring observability a step closer to full observability by approaching the problem in different dimensions.

10 Future Work

In this chapter, I discuss possible future directions in which I can extend my current research for bridging the observability gap in human-robot teams in both the proximate and remote communication scenarios.

ARROCH, VARIL, and VARMAL: In my ARROCH, VARIL, and VARMAL contributions, we use a simple virtual human perception model, but moving forward, I would like to investigate the rich literature of human psychology to motivate the design of complex human-like perception behaviors. Leveraging the knowledge of human psychology, we can also investigate how different types of visualizations, their colors, and sizes, affect the attention span of humans to model a realistic virtual human for generating our human-robot collaborative behaviors.

Another dimension I want to investigate is how AR can be leveraged for human-robot collaborative tasks, when the tasks can be transferable because, in our current studies, we assume that the tasks are non-transferable between human and robot teammates. I am interested to study how AR interfaces can enable human and robot agents to share their intents and feedback when complex collaborative tasks like assembly are carried out in a shared environment with transferable tasks.

One more research direction, I want to explore is the difference in human spatial understanding, user experience, and collaboration efficiency between different types of AR devices like AR headsets, handheld mobile devices, and tablets. Also,

I want to evaluate how the different visualizations affect the human in terms of motion sickness, visual clutter, and the feasibility of using these devices when the human is working on its' own set of tasks while working alongside robotics.

MAARS: In the future, I plan to create a cloud-based architecture of MAARS, enabling the researchers to create many parallel instances of the simulation and allowing them to perform large-scale learning-based experiments. I would like to add more modalities of communication in MAARS including dialogue, and eye gaze, to test different algorithms involving multi-modal communication [188]. Such a cloud-based multi-modal interaction simulator can help researchers working outside the AR field to quickly integrate new modalities in their systems.

Another direction we want to explore is enhancing the perception capabilities. Currently, MAARS simulates monocular cameras for robots, and we want to explore the use cases with 360-degree cameras. This will lead to the simulation of robust virtual humans and robots by integrating object segmentation, detection, and other computer vision techniques that can also support training using reinforcement learning-based algorithms on raw perception data.

Moreover, we want to extend MAARS from the current proximate communication environments and include remote communication, where the human and the virtual robots will be spatially separated. Extension of MAARS to telepresence environments will allow a human operator to control and track a team of robots, and then remotely access the robots to provide help on an as-needed basis. This will open up new avenues to test telepresence systems that can enhance human situational awareness in varied human-robot collaboration scenarios.

GHAL360: I would like to implement and evaluate GHAL360 on a real robot and

conduct studies with human participants for subjective analysis. This will lead to new challenges where the learned visualization policy needs to assist humans in real-time, and the transfer of 360-degree videos requires extensive bandwidth leading to the transfer of low-resolution images in low-bandwidth scenarios. In case of a discrepancy in the visualization policy, I plan to use real-time human feedback to improve the visualization policy.

Currently, the scene analysis component does not go beyond object detection in the current implementation of GHAL360. We would like to study if and how advanced scene analysis methods, such as those based on graph neural networks [189], can contribute to the system. Finally, our current controller suggests the robot to follow the human's intention, which can possibly be improved using a planner to actively guide the human's attention toward the direction of the target object, even when the object is not present in the immediate surroundings.

References

[1] Michael A Goodrich, Alan C Schultz, et al. "Human–robot interaction: a survey". In: *Foundations and Trends® in Human–Computer Interaction* 1.3 (2008), pp. 203–275.

[2] Ronald T Azuma. "A survey of augmented reality". In: *Presence: Teleoperators & Virtual Environments* 6.4 (1997), pp. 355–385.

[3] Zhanat Makhataeva and Huseyin Atakan Varol. "Augmented reality for robotics: a review". In: *Robotics* 9.2 (2020), p. 21.

[4] Michael Gelfond and Yulia Kahl. *Knowledge representation, reasoning, and the design of intelligent agents: The answer-set programming approach*. Cambridge University Press, 2014.

[5] Vladimir Lifschitz. "What Is Answer Set Programming?." In: *AAAI*. Vol. 8. 2008, pp. 1594–1597.

[6] Fangkai Yang et al. "Planning in answer set programming while learning action costs for mobile robots". In: *2014 AAAI Spring Symposium Series*. 2014.

[7] Vladimir Lifschitz. "Answer set programming and plan generation". In: *Artificial Intelligence* 138.1-2 (2002), pp. 39–54.

[8] Esra Erdem, Michael Gelfond, and Nicola Leone. "Applications of answer set programming". In: *AI Magazine* 37.3 (2016), pp. 53–68.

[9] Esra Erdem and Volkan Patoglu. "Applications of ASP in robotics". In: *KI-Künstliche Intelligenz* 32.2-3 (2018), pp. 143–149.

[10] Martin L Puterman. *Markov Decision Processes.: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[11] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. "Planning and acting in partially observable stochastic domains". In: *Artificial intelligence* 101.1-2 (1998), pp. 99–134.

[12] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[13] Brenna D Argall et al. "A survey of robot learning from demonstration". In: *Robotics and autonomous systems* 57.5 (2009), pp. 469–483.

[14] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[15] Christopher JCH Watkins and Peter Dayan. "Q-learning". In: *Machine learning* 8.3-4 (1992), pp. 279–292.

[16] Joyce Y Chai et al. “Collaborative effort towards common ground in situated human-robot dialogue”. In: *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*. ACM. 2014, pp. 33–40.

[17] Jesse Thomason et al. “Learning to interpret natural language commands through human-robot dialog”. In: *Twenty-Fourth International Joint Conference on Artificial Intelligence*. 2015.

[18] Cynthia Matuszek et al. “Learning to parse natural language commands to a robot control system”. In: *Experimental Robotics*. Springer. 2013, pp. 403–415.

[19] Saeid Amiri et al. “Augmenting Knowledge through Statistical, Goal-oriented Human-Robot Dialog”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019.

[20] Stefan Waldherr, Roseli Romero, and Sebastian Thrun. “A Gesture Based Interface for Human-Robot Interaction”. In: *Autonomous Robots* 9.2 (Sept. 2000), pp. 151–173.

[21] Kai Nickel and Rainer Stiefelhagen. “Visual recognition of pointing gestures for human–robot interaction”. In: *Image and vision computing* 25.12 (2007), pp. 1875–1884.

[22] Hee-Deok Yang, A-Yeon Park, and Seong-Whan Lee. “Gesture spotting and recognition for human–robot interaction”. In: *IEEE Transactions on robotics* 23.2 (2007), pp. 256–270.

[23] Tom Williams et al. “Virtual, augmented, and mixed reality for human-robot interaction (vam-hri)”. In: *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2019, pp. 671–672.

[24] Ravi Teja Chadalavada et al. “That’s on my mind! robot to human intention communication through on-board projection on shared floor space”. In: *2015 European Conference on Mobile Robots (ECMR)*. IEEE. 2015, pp. 1–6.

[25] Jongkyeong Park and Gerard Jounghyun Kim. “Robots with Projectors: An Alternative to Anthropomorphic HRI”. In: *Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction*. 2009.

[26] G. Reinhart, W. Vogl, and I. Kresse. “A Projection-based User Interface for Industrial Robots”. In: *IEEE Symposium on Virtual Environments, Human-Computer Interfaces and Measurement Systems*. 2007.

[27] A. Watanabe et al. “Communicating robotic navigational intentions”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015.

[28] Michael Baker et al. “Improved interfaces for human-robot interaction in urban search and rescue”. In: *IEEE International Conference on Systems, Man and Cybernetics*. Vol. 3. 2004, pp. 2960–2965.

[29] M Waleed Kadous, Raymond Ka-Man Sheh, and Claude Sammut. "Effective user interface design for rescue robotics". In: *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*. ACM. 2006, pp. 250–257.

[30] H Kaymaz Keskinpala, Julie A Adams, and Kazuhiko Kawamura. "PDA-based human-robotic interface". In: *2003 IEEE International Conference on Systems, Man and Cybernetics*. Vol. 4. 2003, pp. 3931–3936.

[31] Cynthia Breazeal. "Emotion and sociable humanoid robots". In: *International journal of human-computer studies* 59.1-2 (2003), pp. 119–155.

[32] Kerstin Dautenhahn et al. "How may I serve you?: a robot companion approaching a seated person in a helping context". In: *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*. ACM. 2006, pp. 172–179.

[33] Ben Robins et al. "Robot-mediated joint attention in children with autism: A case study in robot-human interaction". In: *Interaction studies* 5.2 (2004), pp. 161–198.

[34] Cynthia Breazeal et al. "Effects of nonverbal communication on efficiency and robustness in human-robot teamwork". In: *2005 IEEE/RSJ international conference on intelligent robots and systems*. 2005.

[35] Takahiro Miyashita et al. "Haptic communication between humans and robots". In: *Robotics Research*. Springer, 2007, pp. 525–536.

[36] Robert J Stone. "Haptic feedback: A brief history from telepresence to virtual reality". In: *International Workshop on Haptic Human-Computer Interaction*. Springer. 2000, pp. 1–16.

[37] Julie Dumora et al. "Experimental study on haptic communication of a human in a shared human-robot collaborative task". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 5137–5144.

[38] Paul Milgram et al. "Applications of augmented reality for human-robot communication". In: *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93)*. Vol. 3. IEEE. 1993, pp. 1467–1472.

[39] Chih-Wei Chang et al. "Improving the authentic learning experience by integrating robots into the mixed-reality environment". In: *Computers & Education* 55.4 (2010), pp. 1572–1578.

[40] Abraham Prieto Garcia et al. "Mixed reality educational environment for robotics". In: *2011 IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems Proceedings*. IEEE. 2011, pp. 1–6.

[41] Mark Cheli et al. "Towards an augmented reality framework for k-12 robotics education". In: *Proceedings of the 1st International Workshop on Virtual, Augmented, and Mixed Reality for HRI (VAM-HRI)*. 2018.

[42] Z. Zhang et al. “Introducing Reinforcement Learning to K-12 Students with Robots and Augmented Reality”. In: *To appear in proceedings of the International Conference on Robotics in Education (RiE)*. 2023.

[43] Nirit Gavish et al. “Evaluating virtual reality and augmented reality training for industrial maintenance and assembly tasks”. In: *Interactive Learning Environments* 23.6 (2015), pp. 778–798.

[44] Dušan Tatić and Bojan Tešić. “The application of augmented reality technologies for the improvement of occupational safety in an industrial environment”. In: *Computers in Industry* 85 (2017), pp. 1–10.

[45] Ze-Hao Lai et al. “Smart augmented reality instructional system for mechanical assembly towards worker-centered intelligent manufacturing”. In: *Journal of Manufacturing Systems* 55 (2020), pp. 69–81.

[46] Christian Kollatsch et al. “Mobile augmented reality based monitoring of assembly lines”. In: *Procedia Cirp* 23 (2014), pp. 246–251.

[47] Doris Aschenbrenner et al. “Comparing human factors for augmented reality supported single-user and collaborative repair operations of industrial robots”. In: *Frontiers in Robotics and AI* (2019), p. 37.

[48] Dikai Fang et al. “An augmented reality-based method for remote collaborative real-time assistance: from a system perspective”. In: *Mobile Networks and Applications* 25 (2020), pp. 412–425.

[49] Ramsundar Kalpagam Ganesan et al. “Better teaming through visual cues: how projecting imagery in a workspace can improve human-robot collaboration”. In: *IEEE Robotics & Automation Magazine* 25.2 (2018), pp. 59–71.

[50] Wei Wang et al. “Application of augmented reality (AR) technologies in inhouse logistics”. In: *E3S Web of Conferences*. Vol. 145. EDP Sciences. 2020, p. 02018.

[51] Long Qian et al. “A review of augmented reality in robotic-assisted surgery”. In: *IEEE Transactions on Medical Robotics and Bionics* 2.1 (2019), pp. 1–16.

[52] Ashirwad Chowriappa et al. “Augmented-reality-based skills training for robot-assisted urethrovesical anastomosis: a multi-institutional randomised controlled trial”. In: *BJU international* 115.2 (2015), pp. 336–345.

[53] Nuno Costa and Artur Arsenio. “Augmented reality behind the wheel-human interactive assistance by mobile robots”. In: *2015 6th International Conference on Automation, Robotics and Applications (ICARA)*. IEEE, 2015, pp. 63–69.

[54] Long Qian et al. “Augmented reality assisted instrument insertion and tool manipulation for the first assistant in robotic surgery”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 5173–5179.

[55] Ryan M Dickey et al. “Augmented reality assisted surgery: a urologic training tool”. In: *Asian journal of andrology* 18.5 (2016), p. 732.

[56] Francesco Porpiglia et al. “Augmented-reality robot-assisted radical prostatectomy using hyper-accuracy three-dimensional reconstruction (HA 3D™) technology: a radiological and pathological study”. In: *BJU international* 123.5 (2019), pp. 834–845.

[57] Patrick Pessaux et al. “Towards cybernetic surgery: robotic and augmented reality-assisted liver segmentectomy”. In: *Langenbeck’s archives of surgery* 400 (2015), pp. 381–385.

[58] Wen P Liu et al. “Augmented reality and cone beam CT guidance for transoral robotic surgery”. In: *Journal of robotic surgery* 9 (2015), pp. 223–233.

[59] Junchen Wang et al. “Video see-through augmented reality for oral and maxillofacial surgery”. In: *The international journal of medical robotics and computer assisted surgery* 13.2 (2017), e1754.

[60] Chaozheng Zhou et al. “Robot-assisted surgery for mandibular angle split osteotomy using augmented reality: preliminary results on clinical animal experiment”. In: *Aesthetic plastic surgery* 41 (2017), pp. 1228–1236.

[61] Karthik Madhavan et al. “Augmented-reality integrated robotics in neurosurgery: are we there yet?” In: *Neurosurgical focus* 42.5 (2017), E3.

[62] Long Qian, Anton Deguet, and Peter Kazanzides. “ARssist: augmented reality on a head-mounted display for the first assistant in robotic surgery”. In: *Healthcare technology letters* 5.5 (2018), pp. 194–200.

[63] P. Milgram et al. “Applications of augmented reality for human-robot communication”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 1993.

[64] Scott A Green et al. “Evaluating the augmented reality human-robot collaboration system”. In: *International journal of intelligent systems technologies and applications* 8.1-4 (2010), pp. 130–143.

[65] Michael Walker et al. “Communicating Robot Motion Intent with Augmented Reality”. In: *Proceedings of the International Conference on Human-Robot Interaction*. 2018.

[66] Hooman Hedayati, Michael Walker, and Daniel Szafir. “Improving collocated robot teleoperation with augmented reality”. In: *ACM/IEEE International Conference on Human-Robot Interaction*. 2018.

[67] Heni Ben Amor et al. “Intention projection for human-robot collaboration with mixed reality cues”. In: *International Workshop on Virtual, Augmented, and Mixed Reality for HRI (VAM-HRI)*. 2018.

[68] Connor Brooks and Daniel Szafir. “Visualization of Intended Assistance for Acceptance of Shared Control”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020.

[69] Eric Rosen et al. “Communicating and controlling robot arm motion intent through mixed-reality head-mounted displays”. In: *The International Journal of Robotics Research* 38.12-13 (2019), pp. 1513–1526.

[70] Faizan Muhammad et al. “Creating a Shared Reality with Robots”. In: *Proceedings of the 14th ACM/IEEE International Conference on Human-Robot Interaction*. 2019.

[71] Aaquib Tabrez, Matthew B Luebbers, and Bradley Hayes. “Descriptive and Prescriptive Visual Guidance to Improve Shared Situational Awareness in Human-Robot Teaming”. In: *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*. 2022, pp. 1256–1264.

[72] Danny Zhu and Manuela Veloso. “Virtually adapted reality and algorithm visualization for autonomous robots”. In: *Robot World Cup*. Springer. 2016, pp. 452–464.

[73] Shuwen Qiu et al. “Human-robot interaction in a shared augmented reality workspace”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 11413–11418.

[74] Michael Walker et al. “Virtual, augmented, and mixed reality for human-robot interaction: A survey and virtual design element taxonomy”. In: *arXiv preprint arXiv:2202.11249* (2022).

[75] Thomas R Groechel et al. “Reimagining RViz: Multidimensional Augmented Reality Robot Signal Design”. In: *2022 31st IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE. 2022, pp. 1224–1231.

[76] Andre Cleaver et al. “Dynamic Path Visualization for Human-Robot Collaboration”. In: *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*. 2021, pp. 339–343.

[77] Ruth Rosenholtz, Yuanzhen Li, and Lisa Nakano. “Measuring visual clutter”. In: *Journal of vision* 7.2 (2007), pp. 17–17.

[78] Simon Julier et al. “Information filtering for mobile augmented reality”. In: *Proceedings IEEE and ACM International Symposium on Augmented Reality (ISAR 2000)*. IEEE. 2000, pp. 3–11.

[79] Stephen D Peterson, Magnus Axholt, and Stephen R Ellis. “Label segregation by remapping stereoscopic depth in far-field augmented reality”. In: *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society. 2008, pp. 143–152.

[80] Ronald Azuma and Chris Furmanski. “Evaluating label placement for augmented reality view management”. In: *Proceedings of the 2nd IEEE/ACM international Symposium on Mixed and Augmented Reality*. IEEE Computer Society. 2003, p. 66.

[81] Fan Zhang and Hanqiu Sun. “Dynamic labeling management in virtual and augmented environments”. In: *Ninth International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*. IEEE. 2005, 6–pp.

[82] Pieter Abbeel and Andrew Y Ng. “Apprenticeship learning via inverse reinforcement learning”. In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 1.

[83] J Bagnell et al. "Boosting structured prediction for imitation learning". In: *Advances in Neural Information Processing Systems* 19 (2006).

[84] David Silver, James Bagnell, and Anthony Stentz. "High performance outdoor navigation from overhead data using imitation learning". In: *Robotics: Science and Systems IV, Zurich, Switzerland* 1 (2008).

[85] Jiakai Zhang and Kyunghyun Cho. "Query-efficient imitation learning for end-to-end autonomous driving". In: *arXiv preprint arXiv:1605.06450* (2016).

[86] Harish Ravichandar et al. "Recent advances in robot learning from demonstration". In: *Annual Review of Control, Robotics, and Autonomous Systems* 3 (2020), pp. 297–330.

[87] Matthew Luebbers et al. "ARC-LfD: Using Augmented Reality for Interactive Long-Term Robot Skill Maintenance via Constrained Learning from Demonstration". In: *In 2021 International Conference on Robotics and Automation (ICRA)* (2021).

[88] Yohei Hayamizu et al. "Guiding robot exploration in reinforcement learning via automated planning". In: *Proceedings of the International Conference on Automated Planning and Scheduling*. Vol. 31. 2021, pp. 625–633.

[89] Peggy Fidelman and Peter Stone. "Learning ball acquisition on a physical robot". In: *2004 International Symposium on Robotics and Automation (ISRA)*. 2004, p. 6.

[90] Ashutosh Saxena, Lawson LS Wong, and Andrew Y Ng. "Learning grasp strategies with partial shape information." In: *AAAI*. Vol. 3. 2. 2008, pp. 1491–1494.

[91] Minija Tamosiunaite et al. "Learning to pour with a robot arm combining goal and shape learning for dynamic movement primitives". In: *Robotics and Autonomous Systems* 59.11 (2011), pp. 910–922.

[92] Ye Gu, Anand Thobbi, and Weihua Sheng. "Human-robot collaborative manipulation through imitation and reinforcement learning". In: *2011 IEEE International Conference on Information and Automation*. IEEE. 2011, pp. 151–156.

[93] Fotios Dimeas and Nikos Aspragathos. "Reinforcement learning of variable admittance control for human-robot co-manipulation". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 1011–1016.

[94] Konstantinos Tsiakas et al. "An interactive multisensing framework for personalized human robot collaboration and assistive training using reinforcement learning". In: *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments*. ACM. 2017, pp. 423–427.

[95] Matthew B Luebbers et al. "Arc-lfd: Using augmented reality for interactive long-term robot skill maintenance via constrained learning from demonstration". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 3794–3800.

[96] Carl Mueller, Aaquib Tabrez, and Bradley Hayes. “Interactive Constrained Learning from Demonstration Using Visual Robot Behavior Counterfactuals”. In: *Proceedings of the Accessibility of Robot Programming and Work of the Future Workshop at RSS*. Vol. 2021. 2021.

[97] Fouad Sukkar et al. “Guided Learning from Demonstration for Robust Transferability”. In: *arXiv preprint arXiv:2302.03901* (2023).

[98] Andre Cleaver and Jivko Sinapov. “Helping Humans Become Better Teachers for Robots with Augmented Reality”. In: *International Workshop on Virtual, Augmented, and Mixed Reality for HRI (VAM-HRI)*. 2023.

[99] Andreas Blank et al. “Augmented Virtuality Input Demonstration Refinement Improving Hybrid Manipulation Learning for Bin Picking”. In: *Flexible Automation and Intelligent Manufacturing: The Human-Data-Technology Nexus: Proceedings of FAIM 2022, June 19–23, 2022, Detroit, Michigan, USA*. Springer, 2022, pp. 332–341.

[100] Alexander E Kaplan et al. “An Internet accessible telepresence”. In: *Multimedia systems* 5.2 (1997), pp. 140–144.

[101] Rahul Agarwal et al. “The RoboConsultant: telementoring and remote presence in the operating room during minimally invasive urologic surgeries using a novel mobile robotic interface”. In: *Urology* 70.5 (2007), pp. 970–974.

[102] Eric Paulos and John Canny. “Social tele-embodiment: Understanding presence”. In: *Autonomous Robots* 11.1 (2001), pp. 87–95.

[103] Hideaki Kuzuoka et al. “GestureMan: a mobile robot that embodies a remote instructor’s actions”. In: *Proceedings of the ACM conference on Computer supported cooperative work*. 2000, pp. 155–162.

[104] Mark T Bolas and Scott S Fisher. “Head-coupled remote stereoscopic camera system for telepresence applications”. In: *Stereoscopic Displays and Applications*. Vol. 1256. 1990, pp. 113–123.

[105] Tamay Aykut et al. “Delay compensation for actuated stereoscopic 360 degree telepresence systems with probabilistic head motion prediction”. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2018, pp. 2010–2018.

[106] Sei Ikeda, Tomokazu Sato, and Naokazu Yokoya. “High-resolution panoramic movie generation from video streams acquired by an omnidirectional multicamera system”. In: *Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*. 2003, pp. 155–160.

[107] Mojtaba Karimi, Tamay Aykut, and Eckehard Steinbach. “MAVI: A Research Platform for Telepresence and Teleoperation”. In: *arXiv preprint arXiv:1805.09447* (2018).

[108] Kimiya Yamaashi et al. “Beating the Limitations of Camera-Monitor Mediated Telepresence with Extra Eyes”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1996, pp. 50–57.

[109] Gordon M Mair. "Telepresence-the technology and its economic and social implications". In: *1997 International Symposium on Technology and Society Technology and Society at a Time of Sweeping Change. Proceedings*. 1997, pp. 118–124.

[110] Yasamin Heshmat et al. "Geocaching with a beam: Shared outdoor activities through a telepresence robot with 360 degree viewing". In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM. 2018.

[111] John Paulin Hansen et al. "Head and gaze control of a telepresence robot with an HMD". In: *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*. 2018, pp. 1–3.

[112] Jingxin Zhang et al. "A 360 video-based robot platform for telepresent redirected walking". In: *Proceedings of the 1st International Workshop on Virtual, Augmented, and Mixed Reality for Human-Robot Interactions (VAM-HRI)*. 2018, pp. 58–62.

[113] Yeonju Oh et al. "360 vr based robot teleoperation interface for virtual tour". In: *Proceedings of the 1st International Workshop on Virtual, Augmented, and Mixed Reality for HRI (VAM-HRI)*. 2018.

[114] Chao Zhou et al. "AdaP-360: User-Adaptive Area-of-Focus Projections for Bandwidth-Efficient 360-Degree Video Streaming". In: *Proceedings of the 28th ACM International Conference on Multimedia*. 2020, pp. 3715–3723.

[115] Mohammad Hosseini and Viswanathan Swaminathan. "Adaptive 360 VR video streaming based on MPEG-DASH SRD". In: *2016 IEEE International Symposium on Multimedia (ISM)*. IEEE. 2016, pp. 407–408.

[116] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. "Active vision". In: *International journal of computer vision* 1.4 (1988), pp. 333–356.

[117] Alexander Andreopoulos and John K Tsotsos. "50 years of object recognition: Directions forward". In: *Computer vision and image understanding* 117.8 (2013), pp. 827–891.

[118] Sven J Dickinson et al. "Active object recognition integrating attention and viewpoint control". In: *Computer vision and image understanding* 67.3 (1997), pp. 239–260.

[119] Dinesh Jayaraman and Kristen Grauman. "End-to-end policy learning for active visual categorization". In: *IEEE transactions on pattern analysis and machine intelligence* 41.7 (2018), pp. 1601–1614.

[120] Yu-Chuan Su, Dinesh Jayaraman, and Kristen Grauman. "Pano2Vid: Automatic Cinematography for Watching 360 Videos". In: *Asian Conference on Computer Vision*. Springer. 2016, pp. 154–171.

[121] Yu-Chuan Su and Kristen Grauman. "Making 360 video watchable in 2d: Learning videography for click free viewing". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1368–1376.

[122] Nathan Koenig and Andrew Howard. "Design and use paradigms for gazebo, an open-source multi-robot simulator". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2004.

[123] Morgan Quigley et al. “ROS: an open-source Robot Operating System”. In: *ICRA workshop on open source software*. 2009.

[124] Chuang Gan et al. “Threedworld: A platform for interactive multi-modal physical simulation”. In: *arXiv preprint arXiv:2007.04954* (2020).

[125] Angel Chang et al. “Matterport3d: Learning from rgb-d data in indoor environments”. In: *arXiv preprint arXiv:1709.06158* (2017).

[126] Shuran Song et al. “Semantic scene completion from a single depth image”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1746–1754.

[127] Simon Brodeur et al. “Home: A household multimodal environment”. In: *arXiv preprint arXiv:1711.11017* (2017).

[128] Manolis Savva et al. “MINOS: Multimodal indoor simulator for navigation in complex environments”. In: *arXiv preprint arXiv:1712.03931* (2017).

[129] Fei Xia et al. “Interactive gibson benchmark: A benchmark for interactive navigation in cluttered environments”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 713–720.

[130] Claudia Yan et al. “CHALET: Cornell house agent learning environment”. In: *arXiv preprint arXiv:1801.07357* (2018).

[131] Xavier Puig et al. “Virtualhome: Simulating household activities via programs”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8494–8502.

[132] Eric Kolve et al. “Ai2-thor: An interactive 3d environment for visual ai”. In: *arXiv preprint arXiv:1712.05474* (2017).

[133] Fanbo Xiang et al. “Sapien: A simulated part-based interactive environment”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11097–11107.

[134] Xiaofeng Gao et al. “Vrkitchen: an interactive 3d virtual environment for task-oriented learning”. In: *arXiv preprint arXiv:1903.05757* (2019).

[135] Andrew Szot et al. “Habitat 2.0: Training home assistants to rearrange their habitat”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 251–266.

[136] Manolis Savva et al. “Habitat: A platform for embodied ai research”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 9339–9347.

[137] Tetsunari Inamura and Yoshiaki Mizuchi. “SIGVerse: A cloud-based VR platform for research on social and embodied human-robot interaction”. In: *arXiv preprint arXiv:2005.00825* (2020).

[138] Federica Bazzano et al. “Immersive virtual reality-based simulation to support the design of natural human-robot interfaces for service robotic applications”. In: *Augmented Reality, Virtual Reality, and Computer Graphics: Third International Conference, AVR 2016, Lecce, Italy, June 15-18, 2016. Proceedings, Part I 3*. Springer. 2016, pp. 33–51.

[139] Andre Cleaver et al. “HAVEN: a unity-based virtual robot environment to showcase HRI-based augmented reality”. In: *arXiv preprint arXiv:2011.03464* (2020).

[140] Qiaozi Gao et al. “Alexa Arena: A User-Centric Interactive Platform for Embodied AI”. In: *arXiv preprint arXiv:2303.01586* (2023).

[141] Peter R Wurman, Raffaello D’Andrea, and Mick Mountz. “Coordinating hundreds of cooperative, autonomous vehicles in warehouses”. In: *AI magazine* 29.1 (2008), p. 9.

[142] Stanislav Hristov Ivanov, Craig Webster, and Katerina Berezina. “Adoption of robots and service automation by tourism and hospitality companies”. In: *Revista Turismo & Desenvolvimento* 27.28 (2017), pp. 1501–1517.

[143] Ronald Azuma et al. *Recent advances in augmented reality*. Tech. rep. Navel Research Lab, 2001.

[144] Scott A Green et al. “Augmented Reality for Human-Robot Collaboration”. In: *Human Robot Interaction*. 2007.

[145] Matthew B Luebbers et al. “Augmented reality interface for constrained learning from demonstration”. In: *Proceedings of the 1st International Workshop on Virtual, Augmented, and Mixed Reality for HRI (VAM-HRI)*. 2018.

[146] Arthur Juliani et al. *Unity: A General Platform for Intelligent Agents*. 2020. arXiv: `1809.02627 [cs.LG]`.

[147] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning: theory and practice*. Elsevier, 2004.

[148] Alfonso E Gerevini et al. “Deterministic planning in the fifth international planning competition: PDDL3 and experimental evaluation of the planners”. In: *Artificial Intelligence* 173.5-6 (2009), pp. 619–668.

[149] Shiqi Zhang et al. “Mobile Robot Planning Using Action Language BC with an Abstraction Hierarchy”. In: *International Conference on Logic Programming and Nonmonotonic Reasoning*. Springer. 2015, pp. 502–516.

[150] Chitta Baral. *Knowledge representation, reasoning and declarative problem solving*. Cambridge university press, 2003.

[151] Guni Sharon et al. “Conflict-based search for optimal multi-agent pathfinding”. In: *Artificial Intelligence* 219 (2015), pp. 40–66.

[152] Yuqian Jiang et al. “Multi-robot planning with conflicts and synergies”. In: *Autonomous Robots* (2019).

[153] Stefanie Tellex et al. “Understanding natural language commands for robotic navigation and mobile manipulation”. In: *Twenty-Fifth AAAI Conference on Artificial Intelligence*. 2011.

[154] Shiqi Zhang and Peter Stone. “CORPP: Commonsense reasoning and probabilistic planning, as applied to dialog with a mobile robot”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 29. 1. 2015.

[155] Scott A Green et al. “Human-robot collaboration: A literature review and augmented reality approach in design”. In: *International journal of advanced robotic systems* 5.1 (2008), p. 1.

[156] Benjamin J Dixon et al. "Surgeons blinded by enhanced navigation: the effect of augmented reality on attention". In: *Surgical endoscopy* 27.2 (2013), pp. 454–461.

[157] Kishan Chandan, Jack Albertson, and Shiqi Zhang. "Augmented reality visualizations using imitation learning for collaborative warehouse robots". In: *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*. 2022, pp. 1566–1568.

[158] Stefan Schaal. "Is imitation learning the route to humanoid robots?" In: *Trends in cognitive sciences* 3.6 (1999), pp. 233–242.

[159] Tabish Rashid et al. "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 4295–4304.

[160] Ítalo Renan da Costa Barros and Tiago Pereira Nascimento. "Robotic mobile fulfillment systems: A survey on recent developments and research opportunities". In: *Robotics and Autonomous Systems* 137 (2021), p. 103729.

[161] Roni Stern et al. "Multi-agent pathfinding: Definitions, variants, and benchmarks". In: *Twelfth Annual Symposium on Combinatorial Search*. 2019.

[162] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. "A reduction of imitation learning and structured prediction to no-regret online learning". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2011, pp. 627–635.

[163] Kishan Chandan et al. "ARROCH: Augmented Reality for Robots Collaborating with a Human". In: *In 2021 International Conference on Robotics and Automation (ICRA)* (2021).

[164] Kishan Chandan, Xiang Li, and Shiqi Zhang. "Negotiation-based Human-Robot Collaboration via Augmented Reality". In: *The AAAI 2019 Fall Symposium on Artificial Intelligence for Human-Robot Interaction (AI-HRI)*. 2019.

[165] Marcelo Kallmann and Mubbasir Kapadia. "Navigation meshes and real-time dynamic planning for virtual worlds". In: *ACM SIGGRAPH 2014 Courses*. 2014, pp. 1–81.

[166] Alina Beygelzimer et al. "Error limiting reductions between classification tasks". In: *Proceedings of the 22nd international conference on Machine learning*. 2005, pp. 49–56.

[167] Kishan Dhananjay Chandan, Jack Albertson, and Shiqi Zhang. "Learning Visualization Policies of Augmented Reality for Human-Robot Collaboration". In: *Conference on Robot Learning*. PMLR. 2023, pp. 1233–1243.

[168] M. Minsky. "Telepresence". In: *OMNI Magazine* (1980).

[169] Annica Kristoffersson, Silvia Coradeschi, and Amy Loutfi. "A review of mobile robotic telepresence". In: *Advances in Human-Computer Interaction* (2013).

[170] Munjal Desai et al. “Essential features of telepresence robots”. In: *2011 IEEE Conference on Technologies for Practical Robot Applications*. 2011, pp. 15–20.

[171] Leila Takayama and Janet Go. “Mixing metaphors in mobile remote presence”. In: *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*. 2012, pp. 495–504.

[172] Carman Neustaedter et al. “To Beam or not to Beam: A study of remote telepresence attendance at an academic conference”. In: *Proceedings of the 19th ACM conference on computer-supported cooperative work & social computing*. ACM. 2016, pp. 418–431.

[173] Irene Rae and Carman Neustaedter. “Robotic telepresence at scale”. In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM. 2017, pp. 313–324.

[174] Amedeo Cesta et al. “Enabling social interaction through embodiment in ExCITE”. In: *ForItAAL: Second Italian Forum on Ambient Assisted Living*. 2010, pp. 1–7.

[175] Alessandra Maria Sabelli, Takayuki Kanda, and Norihiro Hagita. “A conversational robot in an elderly care center: an ethnographic study”. In: *2011 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2011, pp. 37–44.

[176] Veronica Ahumada-Newhart and Judith S Olson. “Going to school on a robot: Robot and user interface design features that matter”. In: *ACM Transactions on Computer-Human Interaction (TOCHI)* (2019), pp. 1–28.

[177] Irene Rae et al. “A framework for understanding and designing telepresence”. In: *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*. ACM. 2015, pp. 1552–1566.

[178] Theodore C Ruch and John F Fulton. “Medical physiology and biophysics”. In: *Academic Medicine* 35.11 (1960).

[179] Michael Papinutto et al. “The Facespan—the perceptual span for face recognition”. In: *Journal of vision* 17.5 (2017), pp. 16–16.

[180] Vamsidhar Reddy Gaddam et al. “Tiling in interactive panoramic video: Approaches and evaluation”. In: *IEEE Transactions on Multimedia* 18.9 (2016), pp. 1819–1831.

[181] Joseph Redmon and Ali Farhadi. “YOLOv3: An Incremental Improvement”. In: *arXiv* (2018).

[182] Kishan Chandan et al. “Guided 360-Degree Visual Perception for Mobile Telepresence Robots”. In: *Proceedings of the RSS—2020 Workshop on Closing the Academia to Real-World Gap in Service Robotics, Corvallis, OR, USA*. Vol. 13. 2020.

[183] Kishan Chandan et al. “Learning to guide human attention on mobile telepresence robots with 360 vision”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 5297–5304.

[184] Daniel J Simons and Christopher F Chabris. "Gorillas in our midst: Sustained inattentional blindness for dynamic events". In: *perception* 28.9 (1999), pp. 1059–1074.

[185] Tom Williams et al. "Virtual, Augmented, and Mixed Reality for Human-Robot Interaction". In: *Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*. ACM. 2018, pp. 403–404.

[186] James F Mullen et al. "Communicating inferred goals with passive augmented reality and active haptic feedback". In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 8522–8529.

[187] Jeffrey Delmerico et al. "Spatial computing and intuitive interaction: Bringing mixed reality and robotics together". In: *IEEE Robotics & Automation Magazine* 29.1 (2022), pp. 45–57.

[188] Chelsea Zou et al. "ARDIE: AR, Dialogue, and Eye Gaze Policies for Human-Robot Collaboration". In: *arXiv preprint arXiv:2305.04685* (2023).

[189] Saeid Amiri, Kishan Chandan, and Shiqi Zhang. "Reasoning with scene graphs for robot planning under partial observability". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 5560–5567.