# Negotiation-based Human-Robot Collaboration via Augmented Reality

Kishan Chandan[1], Vidisha Kudalkar[1], Xiang Li[2], Shiqi Zhang[1]

*Abstract*— **Effective human-robot collaboration (HRC) requires extensive communication among the human and robot teammates, because their actions can potentially produce conflicts, synergies, or both. We develop a novel augmented reality (AR) interface to bridge the communication gap between human and robot teammates. Building on our AR interface, we develop an AR-mediated, negotiation-based (ARN) framework for HRC. We have conducted experiments both in simulation and on real robots in an office environment, where multiple mobile robots work on delivery tasks. The robots could not complete the tasks on their own, but sometimes need help from their human teammate, rendering human-robot collaboration necessary. Results suggest that ARN significantly reduced the human-robot team's task completion time compared to a non-AR baseline approach.**

## I. INTRODUCTION

Robots are increasingly ubiquitous in everyday environments, but few of them collaborate or even communicate with people in their work time. For instance, Amazon's warehouse robots and people have completely separated work zones [1]. Savioke's Relay robots, having completed hundreds of thousands of deliveries in indoor environments, such as hotels and hospitals, do not interact with people until the moment of delivery [2]. Despite the significant achievements in multi-agent systems [3], human-robot collaboration (HRC), as a kind of multi-agent system, is still rare in practice.

Augmented Reality (AR) focuses on overlaying information in an augmented layer over the real environment to make the objects interactive [4]. On the one hand, AR has promising applications in robotics, and people can visualize the state of the robot in a visually enhanced form while giving feedback at runtime [5]. On the other hand, there are a number of collaboration algorithms developed for multiagent systems (MAS) [3], [6], where a human-robot team is a kind of MAS. Despite the existing research on AR in robotics and multiagent systems, very few have leveraged AR for HRC (see Section II). In this work, we develop an augmented reality-mediated, negotiation-based (**ARN**) framework for HRC problems, where ARN for the first time enables spatially-distant, human-robot teammates to iteratively communicate preferences and constraints toward effective collaborations.

The AR interface of ARN enables the human teammate to visualize the robots' current status (e.g., their current locations) as well as the planned motion trajectories. For instance, *a human user might "see through" a heavy door (via AR) and find a robot waiting for him/her to open the door.*

[1]Department of Computer Science, SUNY Binghamton. Email: {kchanda2, vkudalk1, zhangs}@binghamton.edu
[2]OPPO US Research Center. Email: xiangli.sky@gmail.com

Moreover, ARN also supports people giving feedback to the robots' current plans. For instance, if the user is too busy to help on the door, he/she can indicate "*I cannot open the door for you in three minutes*" using ARN. Accordingly, the robots will incorporate such human feedback for re-planning, and see if it makes sense to work on something else and come back after three minutes. The AR interface is particularly useful in environments with challenging visibility, such as the indoor environments of offices, warehouses, and homes, because the human might frequently find it impossible to directly observe the robots' status due to occlusions.

ARN has been implemented both in simulation and with human participants, where we used a human-robot collaborative delivery task for evaluation. Both human and robots are assigned non-transferable tasks. Results from the real-world experiment suggest that ARN significantly improves the efficiency of human-robot collaboration, in comparison to traditional non-AR baselines. Additionally, we carried out experiments in simulation to evaluate two configurations of ARN (with and without human feedback). From the results, we observe the capability of taking human feedback significantly improved the efficiency of human-robot collaboration, in comparison to the "no feedback" configuration.

## II. RELATED WORK

When humans and robots work in a shared environment, it is vital that they communicate with each other to avoid conflicts, leverage complementary capabilities, and facilitate the smooth accomplishment of tasks. However, humans and robots prefer different modalities for communication. While humans employ natural language, body language, gestures, written communication, etc., the robots need information in a digital form, e.g., text-based commands. Researchers developed algorithms to bridge the human-robot communication gap using natural language [7], [8], [9], [10] and vision [11], [12], [13]. Despite those successes, AR has its unique advantages 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 [14]. We use an AR interface for human-robot collaboration, where the human can directly visualize and interact with the robots' planned actions.

One way of delivering spatial information related to the local environment is through projecting the robot's state and motion intent to the humans using visual cues [15], [16], [17]. For instance, researchers used a LED projector attached to the robot to show its planned motion trajectory, allowing

the human partner to respond to the robot's plan to avoid possible collisions [18]. While such systems facilitate human-robot communication about spatial information, they have the requirement that the human must be in close proximity to the robot. Also, bidirectional communication is difficult in projection-based systems. We develop our AR-based framework that inherits the benefits of spatial information from the projection-based systems while alleviating the proximity requirement, and enabling bidirectional communication.

Early research on AR-based human-robot interaction (HRI) has enabled a human operator to interactively plan, and optimize robot trajectories [19]. More recently, researchers have developed frameworks to help human operators to visualize the motion-level intentions of unmanned aerial vehicles (UAVs) using AR [20], [21]. In another line of research, people used an AR interface to help humans visualize a robot arm's planned actions in the car assembly tasks [22]. However, the communication of those systems is unidirectional, i.e., their methods only convey the robot's intention to the human but do not support the communication the other way around. Our ARN framework supports bidirectional communication toward effective collaborations.

Most relevant to this paper is a system that supports a human user to visualize the robot's sensory information, and planned trajectories, while allowing the robot to prompt information, as well as ask questions through an AR interface [23], [24]. In comparison to their work, our ARN framework supports human-multi-robot collaboration, where the robots collaborate with both robot and human teammates. More importantly, our robots are equipped with the task (re)planning capability, which enables the robots to respond to the human feedback by adjusting their task completion strategy. Our robots' task planning capability enables negotiation and collaboration behaviors within human multi-robot teams.

In the next two sections, we present the major contributions of this research, including the AR interface for human-robot communication, and a human-robot collaboration framework based on the AR interface.

### III. OUR AR INTERFACE FOR HUMAN-ROBOT COMMUNICATION

We have summarized a sample of existing research on human-robot communication, including their limitations. Toward less ambiguity and higher bandwidth, we develop a novel AR interface for human multi-robot communication, as shown in Fig. 1. Our AR interface, for the first time, enables the human teammate to visualize the robots' current state of the world, and their intended plans, while further allowing the human to give feedback on these plans. The robots can then use this feedback to adjust their task completion strategy, if necessary. Next, we focus on the two most important components of our AR interface, namely *Visualizer* for visualizing robots' current states and intended plans, and *Restrictor* for taking the human feedback for plan improvements.
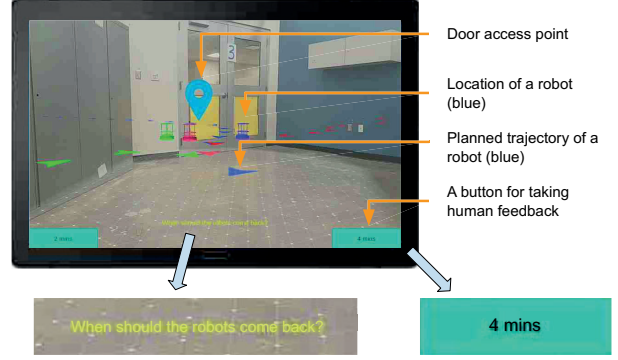


Fig. 1: Our AR interface as displayed on a mobile device. The interface is used for visualizing the robots' locations and planned actions, and taking human feedback using the two interactive buttons at the bottom.

**Visualizer:** The AR interface obtains a set of motion plans ($Trj$), along with the live locations from the robots ($Pose$). The AR interface augments these motion plans as trajectories, and the live locations as visualizable 3D objects (robot avatars) with the $Vslz$ function:

$$V \leftarrow Vslz(Trj, Pose)$$

At every time step $t$, these set of trajectories and robot avatars are rendered as $V$ on the AR interface, where $V$ is a single frame.

**Restrictor:** Once the robots' plans and live locations are augmented, the human can visualize them through the AR device. The AR interface allows the human to also give feedback on the robots' plans. These feedbacks are domain-specific and can involve speech, gestures, interactive buttons, etc. as modes of human input. All the possible feedback is maintained in a feedback library ($F$). The AR interface constantly checks for human feedback, which is then stored in $H$ using the $getFdk$ function, where $H \in F$:

$$H \leftarrow getFdk()$$

If there is no human feedback, then $H$ is $\emptyset$. Human feedback is communicated back to the robots, which can then be used for re-planning if necessary. The human feedback in its original form cannot be used, and hence this human feedback needs to be converted to a set of activated constraints.

$$C \leftarrow Cnsts(H)$$

The $Cnsts$ function takes the human feedback $H$ as input. The output of $Cnsts$ is stored in $C$, where $C$ is a set of activated constraints. The new set of activated constraints can now be used by robots to re-plan if required.

The two components of our AR interface enable the visualization of the robots status and the incorporation of human feedback. In the next section, we describe how we use the AR interface to develop the ARN framework.
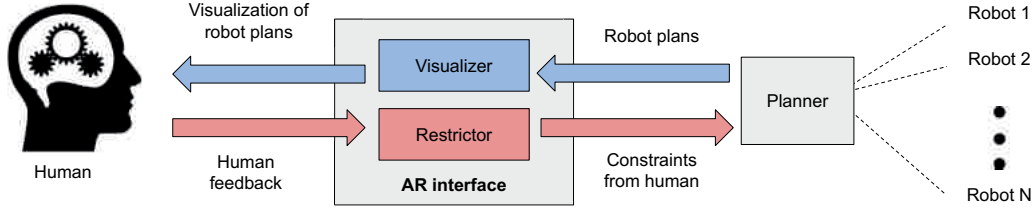
Fig. 2: Key components of our ARN framework: *Visualizer* and *Restrictor* for visualizing robot's intention (for people) and collecting human feedback (for robots) respectively, and *Planner* for computing one action sequence for each robot.

## IV. AR-MEDIATED NEGOTIATION-BASED FRAMEWORK

Multi-agent systems require the agents, including humans and robots, to extensively communicate with each other, because of the inter-dependency among the agents' actions. The inter-dependency can be in the form of state constraints, action synergies, or both. In this section, we describe our augmented reality-mediated, negotiation-based (ARN) framework, as shown in Figure 2, that utilizes the AR interface (Section III) capabilities to enable multi-turn, bidirectional communication between human and robot teammates, and iterative negotiation toward the most effective human-robot collaboration behaviors.

**The ARN Algorithm:** ARN is our general solution that generates plans for all the robots while taking human feedback into account as constraints. The input of Algorithm 1 includes the initial state of $N$ robots ($s$), a set of goal states ($G$), Task Planner $MRP^T$, Motion Planner $MP$, and, $R$ which is a set of $N$ robot teammates.

In Lines 1-3, ARN first initializes three empty lists, $C$, $\textbf{Pose}$, and $\textbf{Trj}$. Then in Line 4, ARN generates a set of initial plans ($\textbf{P}$) for the team of $N$ robots using the task planner ($MRP^T$). Entering a while-loop in Line 5, ARN runs until all robots have completed their tasks. Next, in Line 6, ARN enters a for-loop with $N$ iterations. In Line 7, ARN initially checks if the $i$th robot has reached the goal or not. If not, Lines 8-14 are executed, where ARN first gets the current action ($a_i$) of the $i$th robot using $\textbf{P}$. $MP$ takes $a_i$ as input and generates a motion trajectory, which is then added to $\textbf{Trj}$. In Line 11, ARN gets the pose of the $i$th robot using the $\textbf{getPose}$ function. The poses of all the robots are stored in $\textbf{Pose}$. The if-statement in Line 13 checks if the current action is complete or not, and if it is complete, then the action is popped from the plan queue.

In Line 17, ARN passes the generated set of trajectories and poses to the $\textbf{Vslz}$ function (Section III), which then renders a frame ($V$) on the AR device to augment the motion plans and live locations. Then using $\textbf{getFdk}$ function, ARN checks if any human feedback is available and stores it in $H$(Line 18). Based on $H$, a new set of activated constraints $C'$ are obtained using the $\textbf{Cnsts}$ function (Section III). If the new set of activated constraints are different from the old activated constraints, then ARN generates new plans for the team of robots using the task planner (Line 21), and then the old set of constraints are replaced with the new ones (Line 22). Finally, in Line 24, ARN checks if all the robots

have reached their goals. If the robot(s) have not reached their goals, ARN continues to the next iteration of the while loop. Otherwise, the *break* statement exits the while loop, marking the end of the algorithm (Line 25).

Algorithm 1 enables the bidirectional communication within human-robot teams toward effective human-robot collaboration. Two important steps in ARN include Lines 10 and 21 for motion planning and task planning respectively.

---

**Algorithm 1** ARN Framework

**Input**: $s$, $G$, $MRP^T$, $MP$, $R$

1: Initialize an empty list of activated constraints $C$ as $\emptyset$
2: Initialize an empty list of $\textbf{Pose}$ to store current pose of all robots.
3: Initialize an empty list of size $N$: $\textbf{Trj}$, where $Trj \in \textbf{Trj}$
4: $\textbf{P} = MRP^T(s, \textbf{G}, C)$, where $p \in \textbf{P}$, and $p$ is a queue to store the action sequence of every robot
5: **while** $True$ **do**
6:     **for each** $i \in [0, 1, \cdots, N\text{-}1]$ **do**
7:         **if** $\textbf{goalReached}(s_i, G) == False$ **then**
8:             $a_i \leftarrow \textbf{P}[i].front()$
9:             $R[i]$ executes $a_i$
10:            $\textbf{Trj}[i] \leftarrow MP(a_i)$
11:            $\textbf{Pose}[i] \leftarrow \textbf{getPose}(R[i])$
12:            **if** $a_i$ is completely executed **then**
13:                $\textbf{P}[i].pop()$
14:            **end if**
15:         **end if**
16:     **end for**
17:     $V \leftarrow \textbf{Vslz}(\textbf{Trj}, \textbf{Pose})$ ▷ V is a frame rendered on AR device
18:     $H \leftarrow \textbf{getFdk}()$         ▷ Get feedback from AR device
19:     $C' \leftarrow \textbf{Cnsts}(H)$
20:     **if** $C' \neq C$ **then**
21:         $\textbf{P} \leftarrow MRP^T(s, \textbf{G}, C)$     ▷ Update Plans
22:         $C \leftarrow C'$
23:     **end if**
24:     **if** $s_i$ for all robots $\in G$ **then** ▷ Check if all robots reached goals
25:         break
26:     **else**
27:         continue
28:     **end if**
29: **end while**

---

**Multi-robot Task Planning:** Multi-robot task planning ($MRP^T$) is for computing plans for a team of $N$ robots. The input of $MRP^T$ includes the initial state of $N$ robots ($s$), a set of goal states ($G$), and a set of activated constraints ($C$). $MRP^T$ returns a set of plans $\textbf{P}$, where $p \in \textbf{P}$ consists of a set of actions $[a_1, a_2, \cdots, a_n]$, and $a_i$ is the action for the $i$th robot. Note that $p$ is a queue used for storing a robot's action sequence. $MRP^T$ can be separately developed, and its development is independent of ARN. Our implementation of $MRP^T$ is introduced in Section V.
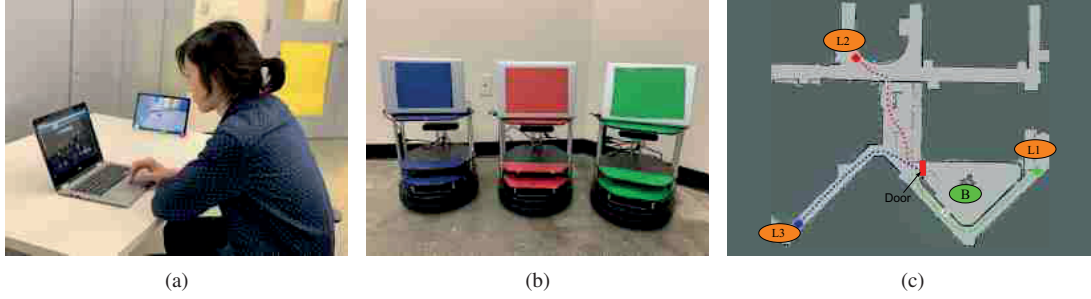
Fig. 3: (a) A human playing the Jigsaw game alongside our AR interface running on a tablet, where the door with yellow paper on it blocks the robots from entering the room by themselves; (b) Turtlebot-2 robot platforms used in the real-world experiments; and (c) Our domain map, including the three "loading" locations ($L1$, $L2$, and $L3$), and base station $B$, where the human participants work on Jigsaw games in the "base station" room and help open the door (in red color) as they feel necessary. The colored trajectories are presented to participants in the non-AR baseline approach.

**Motion Planning:** To visualize the motion plans of the robots, we need to convert the robot actions obtained from $MRP^T$ to a set of trajectories that can be visualized in the AR device. For this purpose, we use a motion planner $MP$, which takes an action $a_i$ as input and returns the trajectory that the robot needs to follow to complete the action.

$$\boldsymbol{Trj} \leftarrow MP(a), \text{ where } Trj = [L_1, L_2, \cdots, L_M]$$

The trajectory is a set of locations, and every location $L_m$ is a set of $x$ and $y$ coordinates for representing a sequence of $M$ 2D coordinates. The $\boldsymbol{getPose}(R[i])$ function returns the current pose of the $i$th robot. The poses of the team of $N$ robots are stored in $\boldsymbol{Pose}$, where $\boldsymbol{Pose}[i]$ represents the pose of the $i$th robot.

In this section, we have described our main contribution ARN for HRC, and how each of the above three functions is used in ARN. But, ARN can be implemented in many ways, and now in the following section, we describe how we implemented ARN in our domain.

## V. MULTI-ROBOT TASK PLANNING

**Single Robot Task Planning:** We use Answer Set Programming (ASP) for robot task planning. ASP is a popular declarative language for knowledge representation and reasoning (KRR) [25], [26], and has been applied to a variety of planning problems [27], [28], [29], including robotics [30]. ASP is particularly useful for robot task planning in domains that include a large number of objects [31]. We formulate five actions in our domain: `approach`, `opendoor`, `gothrough`, `load`, and `unload`. For instance,

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

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

The following shows an example plan for the robot generated by the ASP planner: `approach(D,0).` `opendoor(D,1). gothrough(D,2).`, which suggests

the robot to first approach door `D`, then open door `D`, and finally go through the door at step `I=2`.

**Multi-Robot Task Planning:** Building on this single-robot task planner, we use an algorithm called IIDP (*iterative inter-dependent planning*) to compute joint plans for robot teams [32], where IIDP is a very efficient (though suboptimal) multi-robot planning algorithm. Details are omitted from this paper due to the page limit.

It should be noted that the main contribution of this research is on the AR interface and the ARN framework, and this section is only for completeness.

## VI. EXPERIMENT SETUP AND RESULTS

Experiments have been conducted in simulation and on real robots to evaluate the following two hypotheses in comparison to non-AR baselines: I) ARN improves the overall efficiency in human-robot collaboration, and II) ARN produces better experience for non-professional users.

### A. Real-world Experiments

**Experiment Setup:** Fig. 3(a) shows a human user playing the "Jigsaw" game on a computer, while three Turtlebot-2 robots work on delivering three objects from three different locations to a base station, as shown in Fig. 3(b). The map of this shared office environment is shown in Fig. 3(c). This delivery task requires human-robot collaboration, because both human and robots have their own non-transferable tasks, and that the robots need people to help open a door to "unload" objects to the station. Since the robots do not have an arm, they simply visit places, and loading and unloading actions were not conducted in the real world. All software runs on Robot Operating System (ROS) [33], while door-related functionalities were built on the BWI code base [34].

Eleven participants of ages 20-30 volunteered to participate in the experiment, including four females and seven males.[1] The participants worked in collaboration with the robots to *minimize each individual team member's task-completion time*. It should be noted that this setting covers the goal of minimizing the slowest team member's task

---

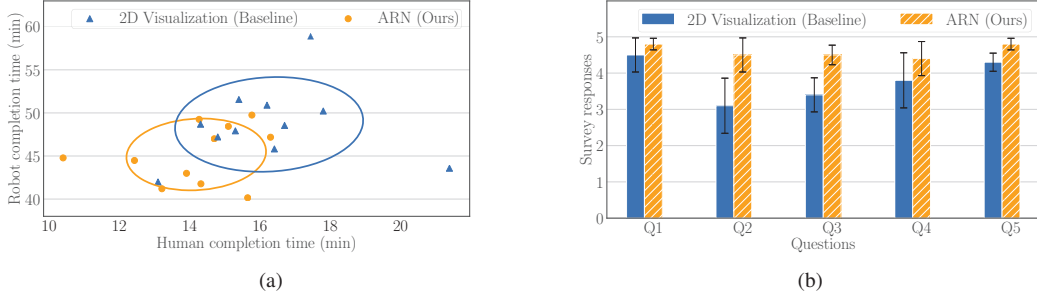[1]The experiments were approved by the Institutional Review Board (IRB).

Fig. 4: (a) Overall performance in task completion time using ARN and a 2D visualization baseline interface (Rviz-based) in human participant study; and (b) Survey responses from participants collaborating with a team of robots on delivery tasks.

completion time, but goes beyond that to capture efficient team members' task completion times.

Each participant used both a baseline system (to be described), and our ARN system to work on the collaborative delivery tasks, where the two systems are randomly ordered to avoid discrepancies in the results caused by people's learning curves. None of the participants have any experience of working with the robots. The Jigsaw puzzles were generated randomly, and no puzzle was repeated throughout the experiment. The settings ensure a fair comparison between the baseline approach and ARN on human-robot collaboration.

**Baseline (2D Visualization):** Without the AR interface, one can use a standard 2D visualization interface to track the status of the robot team. Our non-AR baseline system was built on the visualization tool of Rviz that has been widely used within the ROS community [35]. The participants could see the robots' planned trajectories along with their locations in a 2D map on a laptop, as shown in Fig. 3(c). According to the robots' locations and planned trajectories, the participants decide when to let the robots in through door-opening actions. It should be noted that the participants can potentially find it difficult to map the robots' locations shown on the map to real-world locations. In comparison, the AR interface very well supports the communication of spatial information. For instance, it is highly intuitive to derive the robot's real locations from Fig. 1.

**Results on Overall Task Completion Time:** Fig. 4(a) shows the overall performance of ARN compared to the baseline. The x-axis corresponds to the participants' average task completion time, and the y-axis corresponds to the sum of the three robots' task completion time in total, i.e., $T^{R1} + T^{R2} + T^{R3}$. Each data point corresponds to one trial (using ARN or baseline). The two ellipses show their 2D standard deviations. We see that the data points of ARN are close to the bottom-left corner, which supports Hypothesis-I (ARN improves the overall collaboration efficiency).

Looking into the results reported in Fig. 4(a), we calculated the total time of each trial, i.e.,

$$T^{all} = T^H + T^{R1} + T^{R2} + T^{R3}$$

where $T^H$ and $T^{Ri}$ are the task completion times of the human and the $i$th robot respectively. ARN significantly

improved the performance in $T^{all}$ in comparison to the 2D-visualization baseline, where $p$-value is .011. This shows that ARN performs *significantly* better than the baseline in total task completion time.

**Questionnaires:** At the end of each experiment trial, participants were asked to fill out a survey form indicating their qualitative opinion over 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, thinking, etc.)*; and Q5, *I enjoyed working with the robot and would like to use such a system in the future.* It should be noted that Q1 is a question aiming at confirming if the participants understood the tasks, and is not directly relevant to the evaluation of our hypotheses.

Fig. 4(b) shows the average scores from the human participant surveys. Results show that ARN produced higher scores on Questions Q2-Q5. We also calculated the corresponding $p$-values and observed significant improvements in Q2, Q3, and Q5 with the $p$-values $< .001$. The significant improvements suggest that ARN helps the participants keep track of the robot status, is less distracting, and is more user-friendly to non-professional participants. However, 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 due to the two interfaces being very different by nature.[2]

### B. Simulation Experiments

**Experiment Setup:** Experiments with human participants have shown the effectiveness of ARN. We are interested in how much the "negotiation-based" component contributes to the success of ARN. Considering the practical difficulties in large-scale experiments with human-multi-robot systems, we have built a simulation platform for extensive evaluations. In the simulation, we compared two configurations of ARN: one activated the "Restrictor" for adding constraints from human feedback, and the other did not allow the robots to

---

[2]This submission includes a supplementary demo video for illustrating a complete human-robot collaboration trial.

Fig. 5: (a) Gazebo showing the three robots waiting outside the base station; (b) Enlarged view of the three robots in Gazebo; (c) Enlarged view of virtual AR device showing the three robots waiting outside the base station; and (d) Unity world with the virtual human, and the AR device placed on the table.

take human feedback. The two configurations are referred to as "w/ fdbk" and "w/o fdbk" respectively. Each configuration was evaluated with one hundred trials.

**Simulation of Robot Behaviors:** Fig. 5(a) and Fig. 5(b) present the environment used for simulating robots' navigation behaviors, where we use Gazebo to build the simulation environment [36]. Our simulation environment exactly replicates the real-world environment, where the robots work on the delivery tasks (Fig. 3(c)). We use an implementation of the Monte Carlo Localization algorithm, the *amcl* [37] package of ROS for robot localization [38], and *move base* [39] package of the ROS navigation stack for robot path planning.

**Simulation of AR Interface:** Fig. 5(c) shows the augmented robots waiting at the door in the virtual AR device. The AR interface is simulated using Unity [40]. The virtual AR device is placed on the table and can be rotated left or right, similar to the real-world setup. The virtual AR device supports ARN features, such as visually augmenting the "real world" with the robots' current locations and the planned motion trajectories, and taking human feedback and converting feedback into task constraints.

**Simulation of Human Behaviors:** Human behaviors are simulated, as shown in Fig. 5(d). In each trial, the virtual human is engaged in its own dummy task, while it also has to help the robots to enter the *base station* by opening the door. The door in the simulation environment is not visualizable, but we simulate the door opening actions. The virtual human can take one action at a time from the following six actions: A1, *Work on his/her own task (simplified as staring at the poster in Unity)*; A2, *Tilt the AR device to the left*; A3, *Tilt the AR device to the right*; A4, *Give feedback "I will be busy for two minutes"*; A5, *Give feedback "I will be busy for four minutes"*; and A6. *Open the door*. In the "without feedback" configuration, the virtual human's action space does not include the two feedback actions (A4 and A5).

Prior to every trial, we sample $T^H$ (human's total task completion time) from a Gaussian distribution, where the mean and variance were empirically selected. While the virtual human is "working" on his/her own task, we sample the time of clicking one of "feedback" buttons (A4 or A5) from another Gaussian distribution, where A4 and A5 are randomly selected. The door opening action can be independently triggered by each individual robot that is waiting

outside the door in probability $0.6$, where we sample the door-opening action (to open it or not) every $20$ seconds. Due to the independency, the human is more likely to open the door when there are more robots waiting outside.

After the "human" takes action A4 ("busy for two minutes), the probability of the human opening the door for each robot is significantly reduced to only $0.2$, and this probability is changed back to $0.6$ after the "two minutes" busy time. Finally, when the human is done with his/her own task, this door-opening probability is increased to $0.9$.

TABLE I: Simulation Results

| Method | $T^H$ | $T^{all}$ | $T^{R_{last}}$ |
|---|---|---|---|
| ARN (w/ feedback) | 15.67 (3.04) | 46.64 (4.31) | 16.65 (0.73) |
| ARN (w/o feedback) | 16.77 (4.73) | 50.43 (6.35) | 17.94 (0.90) |

**Results:** Table I shows the average task completion time of the virtual human ($T^H$), the average task completion time of the team of robots ($T^{all}$), and the average task completion time of the last robot ($T^{R_{last}}$). The two rows correspond to the two configurations: ARN with feedback, (i.e., Restrictor being activated), and ARN without feedback (i.e., Restrictor not being activated). The average completion times are lower for all the trials of ARN with feedback, as compared to the "without feedback" configuration. Furthermore, we calculated the total time of each trial, i.e., $T^{all} = T^H + T^{R1} + T^{R2} + T^{R3}$. We observed significant improvements in the overall task completion time ($T^{all}$) with a $p$-value $< .001$ in the comparisons between the two ARN configurations. This shows that the feedback feature of ARN significantly contributes to the effectiveness of ARN in multi-turn negotiation-based collaboration behaviors.

## VII. CONCLUSIONS

In this paper, we introduce a novel augmented reality-mediated, negotiation-based framework, called ARN, for human-robot collaboration tasks. The human and robot teammates work on non-transferable tasks, while the robots have limited capabilities and need human help at certain phases for task completion. ARN enables human-robot negotiations through visualizing robots' current and planned actions, while also incorporating the human feedback into robot re-planning. Experiments in simulation and with human participants show that ARN significantly increased the overall efficiency of human-robot collaboration, in comparison to a non-AR approach from the literature.

## REFERENCES

[1] P. R. Wurman, R. D'Andrea, and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," *AI magazine*, vol. 29, no. 1, p. 9, 2008.

[2] S. H. Ivanov, C. Webster, and K. Berezina, "Adoption of robots and service automation by tourism and hospitality companies," 2017.

[3] M. Wooldridge, *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[4] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. Mac-Intyre, "Recent advances in augmented reality," Navel Research Lab, Tech. Rep., 2001.

[5] S. A. Green, M. Billinghurst, X. Chen, and J. G. Chase, "Augmented reality for human-robot collaboration," in *Human Robot Interaction*, 2007.

[6] P. Stone and M. Veloso, "Multiagent systems: A survey from a machine learning perspective," *Autonomous Robots*, vol. 8, no. 3, pp. 345–383, 2000.

[7] J. Y. Chai, L. She, R. Fang, S. Ottarson, C. Littley, C. Liu, and K. Hanson, "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.

[8] J. Thomason, S. Zhang, R. J. Mooney, and P. Stone, "Learning to interpret natural language commands through human-robot dialog," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[9] C. Matuszek, E. Herbst, L. Zettlemoyer, and D. Fox, "Learning to parse natural language commands to a robot control system," in *Experimental Robotics*. Springer, 2013, pp. 403–415.

[10] S. Amiri, S. Bajracharya, C. Goktolga, J. Thomason, and S. Zhang, "Augmenting knowledge through statistical, goal-oriented human-robot dialog," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[11] S. Waldherr, R. Romero, and S. Thrun, "A gesture based interface for human-robot interaction," *Autonomous Robots*, vol. 9, no. 2, pp. 151–173, Sep 2000. [Online]. Available: https://doi.org/10.1023/A:1008918401478

[12] K. Nickel and R. Stiefelhagen, "Visual recognition of pointing gestures for human–robot interaction," *Image and vision computing*, vol. 25, no. 12, pp. 1875–1884, 2007.

[13] H.-D. Yang, A.-Y. Park, and S.-W. Lee, "Gesture spotting and recognition for human–robot interaction," *IEEE Transactions on robotics*, vol. 23, no. 2, pp. 256–270, 2007.

[14] R. T. Azuma, "A survey of augmented reality," *Presence: Teleoperators & Virtual Environments*, vol. 6, no. 4, pp. 355–385, 1997.

[15] J. Park and G. J. Kim, "Robots with projectors: An alternative to anthropomorphic hri," in *Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction*, 2009.

[16] A. Watanabe, T. Ikeda, Y. Morales, K. Shinozawa, T. Miyashita, and N. Hagita, "Communicating robotic navigational intentions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[17] G. Reinhart, W. Vogl, and I. Kresse, "A projection-based user interface for industrial robots," in *2007 IEEE Symposium on Virtual Environments, Human-Computer Interfaces and Measurement Systems*, June 2007, pp. 67–71.

[18] R. T. Chadalavada, H. Andreasson, R. Krug, and A. J. Lilienthal, "That's on my mind! robot to human intention communication through on-board projection on shared floor space," in *Mobile Robots (ECMR), 2015 European Conference on*. IEEE, 2015, pp. 1–6.

[19] P. Milgram, S. Zhai, D. Drascic, and J. Grodski, "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, July 1993, pp. 1467–1472 vol.3.

[20] M. Walker, H. Hedayati, J. Lee, and D. Szafir, "Communicating robot motion intent with augmented reality," in *Proceedings of the International Conference on Human-Robot Interaction*, 2018.

[21] H. Hedayati, M. Walker, and D. Szafir, "Improving collocated robot teleoperation with augmented reality," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, 2018, pp. 78–86.

[22] H. B. Amor, R. K. Ganesan, Y. Rathore, and H. Ross, "Intention projection for human-robot collaboration with mixed reality cues," in *Proceedings of the 1st International Workshop on Virtual, Augmented, and Mixed Reality for HRI (VAM-HRI)*, 2018.

[23] F. Muhammad, A. Hassan, A. Cleaver, and J. Sinapov, "Creating a shared reality with robots," in *Proceedings of the 14th ACM/IEEE International Conference on Human-Robot Interaction*, 2019.

[24] M. Cheli, J. Sinapov, E. Danahy, and C. Rogers, "Towards an augmented reality framework for k-12 robotics education," in *1st International Workshop on Virtual, Augmented and Mixed Reality for Human-Robot Interaction (VAMHRI)*, 2018.

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

[26] V. Lifschitz, "What is answer set programming?." in *AAAI*, vol. 8, 2008, pp. 1594–1597.

[27] F. Yang, P. Khandelwal, M. Leonetti, and P. H. Stone, "Planning in answer set programming while learning action costs for mobile robots," in *2014 AAAI Spring Symposium Series*, 2014.

[28] V. Lifschitz, "Answer set programming and plan generation," *Artificial Intelligence*, vol. 138, no. 1-2, pp. 39–54, 2002.

[29] E. Erdem, M. Gelfond, and N. Leone, "Applications of answer set programming," *AI Magazine*, vol. 37, no. 3, pp. 53–68, 2016.

[30] E. Erdem and V. Patoglu, "Applications of asp in robotics," *KI-Künstliche Intelligenz*, vol. 32, no. 2-3, pp. 143–149, 2018.

[31] Y. Jiang, S. Zhang, P. Khandelwal, and P. Stone, "Task planning in robotics: an empirical comparison of pddl- and asp-based systems," *Frontiers of Information Technology & Electronic Engineering*, vol. 20, no. 3, pp. 363–373, 2019.

[32] Y. Jiang, H. Yedidsion, S. Zhang, G. Sharon, and P. Stone, "Multi-robot planning with conflicts and synergies," *Autonomous Robots*, pp. 1–22, 2019.

[33] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[34] P. Khandelwal, S. Zhang, J. Sinapov, M. Leonetti, J. Thomason, F. Yang, I. Gori, M. Svetlik, P. Khante, *et al.*, "Bwibots: A platform for bridging the gap between ai and human–robot interaction research," *The International Journal of Robotics Research*, vol. 36, no. 5-7, pp. 635–659, 2017.

[35] "Rviz - ros wiki," http://wiki.ros.org/rviz.

[36] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE, 2004, pp. 2149–2154.

[37] "Amcl - ros wiki," http://wiki.ros.org/amcl.

[38] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 2. IEEE, 1999, pp. 1322–1328.

[39] "Move_base - ros wiki," http://wiki.ros.org/move_base.

[40] "Unity real-time development platform — 3d, 2d vr & ar visualizations," https://unity.com/.