Received 24 January 2023, accepted 1 March 2023, date of publication 6 March 2023, date of current version 14 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3253513

RESEARCH ARTICLE

Acquisition of Inducing Policy in Collaborative Robot Navigation Based on Multiagent Deep Reinforcement Learning

MITSUHIRO KAMEZAKI[1], (Member, IEEE), RYAN ONG[2], AND SHIGEKI SUGANO[2], (Fellow, IEEE)
[1]Waseda Research Institute for Science and Engineering, Waseda University, Shinjuku-ku, Tokyo 162-0044, Japan
[2]Department of Modern Mechanical Engineering, Waseda University, Shinjuku-ku, Tokyo 169-8555, Japan

Corresponding author: Mitsuhiro Kamezaki (kame-mitsu@aoni.waseda.jp)

This work was supported in part by the Japan Science and Technology Agency (JST) PRESTO under Grant JPMJPR1754; in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant 19H01130; and in part by the Waseda Research Institute Science and Engineering, Waseda University.

ABSTRACT To avoid inefficient movement or the freezing problem in crowded environments, we previously proposed a human-aware interactive navigation method that uses inducement, i.e., voice reminders or physical touch. However, the use of inducement largely depends on many factors, including human attributes, task contents, and environmental contexts. Thus, it is unrealistic to pre-design a set of parameters such as the coefficients in the cost function, personal space, and velocity in accordance with the situation. To understand and evaluate if inducement (voice reminder in this study) is effective and how and when it must be used, we propose to comprehend them through multiagent deep reinforcement learning in which the robot voluntarily acquires an inducing policy suitable for the situation. Specifically, we evaluate whether a voice reminder can improve the time to reach the goal by learning when the robot uses it. Results of simulation experiments with four different situations show that the robot could learn inducing policies suited for each situation, and the effectiveness of inducement is greatly improved in more congested and narrow situations.

INDEX TERMS Autonomous mobile robot, multiagent deep reinforcement learning, inducing policy acquisition, collaborative robot navigation.

I. INTRODUCTION

Path planning for autonomous mobile robots has been widely studied for various applications [1]. Robot navigation in dynamic environments is actively investigated, including the velocity obstacle methods [2], social force models [3], and machine learning-based methods [4]. However, these studies focus on using passive-avoidance strategies (PAS) to enable the robot to avoid humans passively, so the robot often gets stuck in crowded and dynamic spaces [5]. To move adequately in such environments, the robot must use an active-inducible strategy (AIS) that can voluntarily make a path for it by encouraging an obstructing human to move. Thus, we previously proposed a human-aware interactive navigation method [6] that uses *inducement*, i.e., voice reminders, e.g., "Let me pass," or physical touch [7], [8]. We then developed a dynamic waypoint navigation (DWN) method [9] as a model-based path planning method and an inducible social force model (*i*SFM) [10] for efficient proximal crowd navigation. We confirmed that these methods enable the robot to move in crowded and dynamic spaces without being stuck by using inducement.

The navigation systems that we previously developed adopt a rule-based deterministic approach, so the robot's behavior is largely affected by parameter settings, e.g., coefficients in the cost function, personal space, velocity, and inducement. So far, we adjusted the above parameters to be suitable for the situations from exploratory experiments. For example, the cost of using physical touch is prettily large so that the robot hardly uses it [10], but the value does not have versatility. In the situation as shown in Fig. 1, the use of inducement depends on many factors, including human

The associate editor coordinating the review of this manuscript and approving it for publication was Zeyang Xia.

attributes, e.g., looking at a painting, task contents, e.g., the need to hurry due to an emergency evacuation, and environmental contexts, e.g., quiet museum, which are difficult for the robot to recognize. Thus, it is unrealistic to pre-define a set of parameters in accordance with the situation.

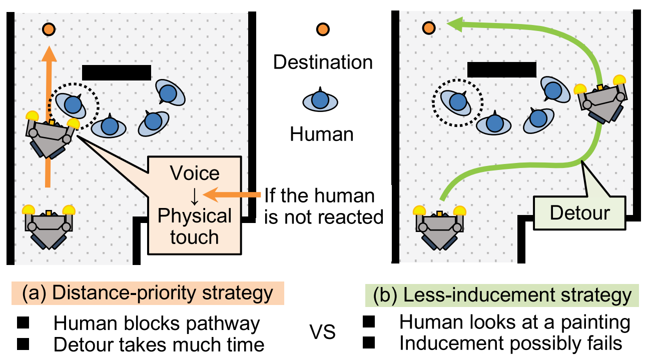


FIGURE 1. **Two typical situation policies; inducement or non-inducement, which basically differ in social accommodation and robot efficiency.**

To understand and evaluate if navigation with active inducement is effective and how and when it must be used, we propose to analyze them through a multiagent reinforcement learning framework in which the robot voluntarily acquires an inducing policy suitable for the situation. Learning systems enable the agent to define its own rules based on the obtained data. To develop a decision-making system for our purpose, multiagent deep reinforcement learning (MDRL) [11] would be effective. Recently, MDRL has been used to learn both the control and strategic aspects in soccer games [12], to learn crowd dynamics and determine efficient paths [13], and to develop an intersection management system for automated vehicles [14]. These studies consider the relationship between agents but do not consider how agents could directly interact with one another to reach their goal nor active inducement. As a similar idea to *inducement*, an approach that incorporates learning communication for multiagent cooperation was proposed [15], [16], [17], [18], [19]. In this approach, agents share or receive *action intention* through a dedicated communication channel, but in our approach, agents can directly use *inducement* as a voluntary action, which enables the robot to acquire a more realistic cooperative inducement policy.

In summary, no learning systems that can evaluate the effectiveness of using inducement exist. We model navigation as a cooperative task that focuses on the convergence of the entire group rather than the individual. We thus modify an MDRL-based navigation method that follows partially observable Markov decision processes [20] by introducing an inducement system. We also measure when and how often inducements are used to examine how the environment affects an agent's desire to use inducement. Our main contributions are:

- To be the first trial to evaluate an inducement strategy in robot navigation through a multiagent deep reinforcement learning framework.
- To evaluate if inducement, i.e., voice reminder, can improve overall efficiency and goal convergence by comparing collective goal convergence times.
- To determine when to use voice reminders by recording their usage frequency and examining inference models.

II. DEVELOPMENT TOOLS AND SETTINGS

In this section, we introduce the tools, frameworks, and network parameters used while training and testing our policies.

A. PRIMARY TOOLS

The primary tools for development are Unity's ML-Agents Toolkit (Release 18), Unity's game engine, and PyTorch (1.10.1). The Unity ML-Agents Toolkit enables Unity to act as an environment for training agents. It includes state-of-the-art algorithms, a Python API for training agents, a gym wrapper, and a built-in inference engine to test trained policies. Unity's engine is used for the simulation environment [21], and PyTorch is the machine learning framework. Unity's scripting API is used to create the random generation of agent positions and spawns and determine an agent's arrival to their goal and their collision with another object.

TABLE 1. **Network settings of MA-POCA trainer.**

Hyper-parameters				Network settings	
Batch size	2048	Epsilon	0.15	Normalize	False
Buffer size	20480	Lambda	0.95	Hidden units	256
Learning rate	0.0002	Epoch	3	Layers	3
Beta	0.003	Learning rate	Constant	Conditioning type	Hyper

Reward signals		
	Extrinsic	Curiosity
Gamma	0.99	0.99
Strength	1.0	0.02*

* For experiment S3, curiosity strength was set to 0.06

B. METHODS

1) TRAINING ALGORITHM

For the training algorithm and architecture, we used Unity's Multi-Agent Posthumous Credit Assignment (MA-POCA), which is a multiagent trainer that trains a centralized critic for a group of agents [22]. The benefit of using MA-POCA is that it enables group rewards alongside individual rewards. Group rewards are imperative when recreating a cooperative task such as path planning since it incentivizes agents to take selfless actions that prioritize the teams' goal over their own individual goals. MA-POCA's novel architecture for COunterfactual Multi-Agent Policy Gradients (COMA) [23] utilizes self-attention in place of a fully connected layer with absorbing states over active agents in the critic network to address issues of posthumous credit assignment. Moreover, MA-POCA enables a variable number of inputs, and the network can compute a counterfactual baseline using self-attention [24]. MA-POCA uses temporal difference [25] to calculate targets for the value function and baseline updates. The above algorithm and architecture use a framework of centralized training and decentralized execution that enables policies to use extra information during training.

2) NETWORK SETTINGS

We used the hyper-parameters as listed in Table 1. The values were basically chosen based on the common configurations for POCA in ML-Agents' training configuration documentation [26], followed by tuning the hyperparameters until policy statistics (value estimates and entropy) were stable. For the network settings, we used 256 hidden units, three layers, and HyperNetworks [27], which is a conditioning type for the policy using goal observations. Alongside our extrinsic rewards, which are explained in the reward function section, we also utilized curiosity [28] for training our agents as our reward policy is sparse.

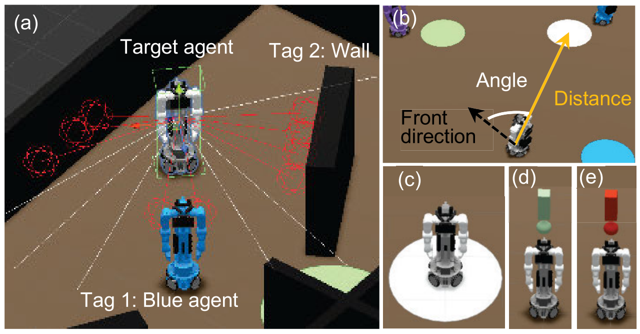


FIGURE 2. Agent model and observations. (a) RayCast perception sensor, (b) relative goal distance and angle, (c) agent on goal state, (d) inducer state, and (e) inducee state.

C. TRAINING AND TESTING

The models were trained on a computer with an NVIDIA RTX 3090 graphics card, an AMD Ryzen 9 5950X 16-Core processor, and 128 GB of RAM. All models were trained in a variant of the scenario with randomly generated agent spawns, agent orientations, and goal spawns to converge with a minimum of 45 million training steps, each episode lasting 30 seconds unless all agents do not crash (1st training). As Unity's engine runs at 60 frames per second and each episode is 1800 steps long, this equates to 25,000 episodes of training. After training the agents on the randomly generated variants, we trained the non-inducement- and inducement-based models equally for 3,000 episodes on the set spawn variant (2nd training). This was a small training duration to avoid overfitting. After training, we ran a final test or inference on the policy for 1,000 episodes and evaluated the group convergence time, collision count, and inducement usage across all 1,000 episodes. The training was executed in parallel with 16 environments and a 20× time scale. Initial training, secondary training, and the inference test took approximately 6 hours, 45 minutes, and 12 minutes, respectively.

TABLE 2. Agent action space.

Action branch	Action	Speed
Linear movement	Forward (↑)	1.4 m/s
	Backward (↓)	0.7 m/s
	Right (→)	0.7 m/s
	Left (←)	0.7 m/s
	No movement	0.0 m/s
Rotation	Clockwise (↻)	$3\pi/4$ rad/s
	Counter-clockwise (↺)	$3\pi/4$ rad/s
	No rotation	0 rad/s
Inducement	Voice reminder	N/A
	No explicit action	N/A

III. IMPLEMENTATION

This section elaborates on the agent model and action space, the types of observations, the environments to evaluate the effectiveness of inducement, and explains how the inducement interaction is rewarded and punished.

A. AGENT MODEL AND ACTION SPACE

1) AGENT MODEL

One of the limitations of using the combination of MA-POCA and group rewards is that it requires all agents in a group to be homogenous. Thus, all agents are identical as they are modeled as a group of agents that try to reach their own goals. All agents, as shown in Fig. 2, are modeled to be human-like in terms of fields of vision and velocity. Each agent has a collider of 0.85 (width) × 0.65 (depth) × 1.90 (height) m that uses a collider and collision Unity classes to check when the agent is either on their goal or colliding with another agent or wall.

2) AGENT ACTION SPACE

All agents feature a discrete action space with three branches, i.e., linear movement, rotation, and inducement, as listed in Table 2. The agent has five possible actions in the linear movement branch, including forward, backward, right, left, and no movement. The forward movement speed was determined based on the average adult human walking speed (1.4 m/s), while the other values were halved as humans typically walk forward and rotate as necessary. The rotation branch has both clockwise and counterclockwise rotations. The inducement enables agents to explicitly interact with other agents. There are no direct velocity implications behind the use of a voice reminder, but agents will be rewarded and punished based on their reactions. To reach the above velocities, the agent selects the respective action repeatedly throughout the 60 frames.

B. OBSERVATIONS

1) RAYCAST OBSERVATION

RayCast observation comes from the Unity ML-Agents toolkit. The RayCast perception sensor has a 200° band

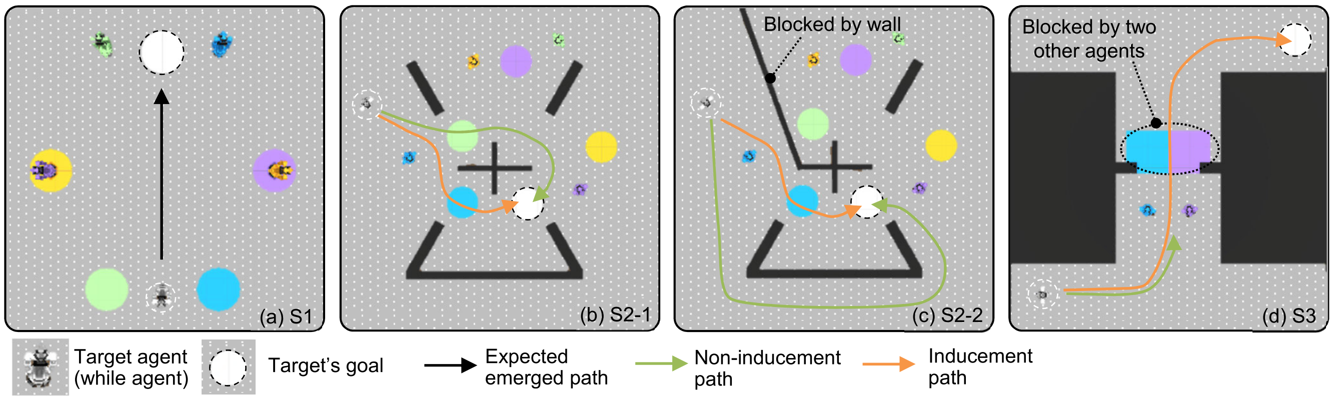


FIGURE 3. Simulation environments (15 × 15 m), which includes five agents, their corresponding goals, and walls. (a) S1: Baseline (open). (b) S2-1: Small detours (similar path distances). (c) S2-2: Large detours (divergent path distances). (d) S3: Convergence (passage blocked by two other agents).

horizontal band of vision to replicate human vision and has 15 equally spread 5-m-long rays, as shown in Fig. 2 (a). The sensors send out a set of rays and output values based on the relative distance between the sensor and the tagged object (e.g., the blue agent is tagged as Tag 1).

2) VECTOR OBSERVATION

There are five vector observations, which are necessary for the agents' ability to learn how to reach their respective goals, use voice reminders, and react to them. The first two vector observations are the agent's relative distance to their goal ([0, 1]) and the agent's relative angle to their goal ([−1, 1]), as shown in Fig. 2 (b). Unlike other models that give absolute positions of both the agents and their goals [29], [30], the agents are only provided relative scalar values, to meet conditions in a real environment. Fig. 2 (c) shows a state when the agent has reached their respective goal.

3) INDUCEMENT OBSERVATION

As shown in Figs. 2 (d) and (e), the agents' inducer and inducee states are represented by green and red exclamations, respectively. The inducer state indicates that the agent has used a voice reminder and initiated the inducement event, while the inducer state indicates that they are a recipient of a voice reminder. Agents are rewarded and punished based on their reaction to an inducement, so the agents must be provided information when they are being induced.

The rationale behind these observations is to replicate human acknowledgment when being addressed. Notably and comparative to real life, the agents are oblivious to the goal position of all the other agent's respective goals.

C. ENVIRONMENT

Our training and test environments have four different situations: a baseline, efficiency (×2), and convergence.

1) BASELINE (S1)

This situation simulates a standard open environment with plenty of space to move around, as shown in Fig. 3 (a). As most conventional multiagent path planning studies focus on this open environment [29], [30], this model will examine the effects inducements can have on the standard setting.

2) EFFICIENCY (S2)

This situation will test the efficiency in two different situations. In the first one, as shown in Fig. 3 (b), the agent is tasked to take the most efficient route when small detours are presented. In the second one, as shown in Fig. 3 (c), the agent is limited to two paths, i.e., an inefficient detour route or the shortest route but blocked by the blue agent.

3) CONVERGENCE (S3)

This situation will simply test the agents' ability to overcome a single passage blocked by two agents by encouraging them to move with voice reminders, as shown in Fig. 3 (d).

D. INDUCEMENT INTERACTION

Inducement is modeled as an event that rewards and punishes agents based on how they react over the inducement duration. Inducement begins whenever an agent uses the voice reminder action and there are other agents within a 2.5-m radius to be induced and lasts 2 seconds, as shown in Fig. 4 (a). Conditions to start the inducement include:

- The agent is not already being induced.
- 8 seconds of voice reminder cooldown is available.

Acceptable values for how often agents can use inducement have a heavily subjective range, but in this study, to evaluate the effectiveness of inducement and its proper use of them, we selected the above cooldown time. Agents cannot use voice reminders when they have already received another agent's inducement, but agents can be induced by multiple agents simultaneously. Once the inducement event begins, the algorithm temporarily logs the relative distance between relevant agents and the distance that the inducee moved during inducement before and after the event. Paired with relative distances between each agent and their respective goals, the reward function determines if the inducer and inducee(s) acted properly. The conditions for successful voice reminders are:

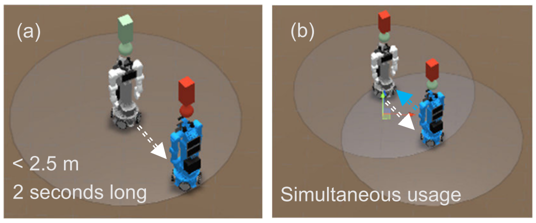


FIGURE 4. Interaction using inducement. (a) standard inducement event and (b) simultaneous usage situation.

- The inducer can move at least 1 m closer toward its goal and the inducer's goal is farther than the recipients.
- At least one other agent must have been part of the inducement event.

The conditions for a successful response are:

- An increase in relative distance between both agents so that the inducer can pass (at least 0.5 m).
- The inducee contributed to the inducement event by moving at least 1 m.

Both agents can be the inducer and inducee when they use voice reminders at the same time, as shown in Fig. 4 (b). This scenario is indicative of real-world interaction. In this case, the agents can obtain the rewards for being both an inducer and inducee, if all requirements are satisfied.

E. REWARD FUNCTION

The reward function is set to focus on group convergence over the individual by prioritizing agents with further goals. The reward and punishments except for those for inducement were set based on the best practices for stable training [31]. The best practices are a max reward magnitude of no greater than 1.0 and a small penalty to help faster convergence.

Table 3 lists the relationship between actions and rewards. Agents are rewarded for each step when they remain on their goal, and the reward is linearly distributed across that time. Specifically, if an agent stays on its goal for 180 frames (3 s), it receives a reward of 0.1, and the max is 1.0 (30 s). Collisions are an immediate penalty. Proper usage of voice reminders rewards the inducer, while poor usage punishes it. The response to voice reminders is also rewarded and punished accordingly. The values for the inducement usage and responses were scaled to be slightly more impactful than the reward for goal arrival since it is inducement to a submissive movement that is away from their goal. The reward for goal arrival at 3 s is 0.1, so the reward for a successful reaction to a voice reminder (2 s) is 0.12 to compensate the inducee for relocating off their goal. Moreover, we set a group punishment for each agent of -0.0055 for every step that all agents are not on their goals to ensure timely group convergence, and the max is -1.0 ($\approx$1800 steps $\times -0.0055$). Ending the punishment requires that all agents are on their goal, so an agent must learn to avoid crashes since group convergence would no longer happen.

IV. EXPERIMENTS

The results are based on the collective group of agents, but the primary purpose of evaluating the effectiveness of inducement is on the white agent as they are presented with the most complex routes, specifically in S2 and S3. The numbers of collisions are those that occur over one episode summary (50 episodes). The success rate was calculated by checking if all agents were on the goal before the end of the episode (30 s).

TABLE 3. Reward function. (I: individual, G: Group, R: Reward, and P: Punishment.)

Action		Reward	I	G	R	P	Description
Goal arrival		1.0	🟦		🟩		Rewarded that agent remains on their goal
Agent collision		1.0	🟦			🟥	Collision with another agent
Wall collision		1.0	🟦			🟥	Collision with a wall
Inducer voice reminder	Effective	0.12	🟦		🟩		Helped agent get to goal
	Poor	0.09	🟦			🟥	Did not help agent or unnecessary
	Useless	0.20	🟦			🟥	Nobody in range
Inducee response	Well received	0.12	🟦		🟩		Moved enough for the agent to pass
	Ignored	0.09	🟦			🟥	Did not move enough
Time penalty		1.0		🟦		🟥	Punished if **ALL** agents are not on their goals

A. OPEN ENVIRONMENT (S1)

We ran three experiments, ideal, non-inducement, and inducement, as shown in Figs. 5 (a)–(c). In ideal, the agents are trained on the set spawn positions for the entire training time to examine the effects of training on the test environment as well as find the lower bound for arrival times. As shown in Fig. 5 (a), the agents learned the trends of all the other agents' movements, which led to a spiral movement toward the goal. This movement is ideal but quite unrealistic for a group of agents to move with such synchronization. Fig. 5 (b) shows a more normal movement with all agents moving directly toward their goal until they were close to one another. Once the agents were close to one another, several agents waited while others continued to move toward their goal. In Fig. 5 (c), while several agents simply avoided the others due to a bountiful amount of space, others used inducement to help get a better path. Fig. 5 (d) shows the average arrival time and the number of collisions. The results show convergences time of 5.3, 7.4, and 8.0 s for ideal, non-inducement, and inducement, respectively. Collisions only occurred twice in the non-inducement and inducement throughout 1000 episodes, and the success rates were 98.4% for non-inducement and 97.9% for inducement.

In summary, the introduction of inducement was not beneficial to agents in an open environment. In fact, it slightly harmed the overall efficiency of the group by 7.9%.

B. EFFICIENCY ENVIRONMENT–SMALL DETOUR (S2-1)

We tested two experiments, inducement and non-inducement. In non-inducement, we found that the agent took multiple routes despite having the same spawn location and goals.

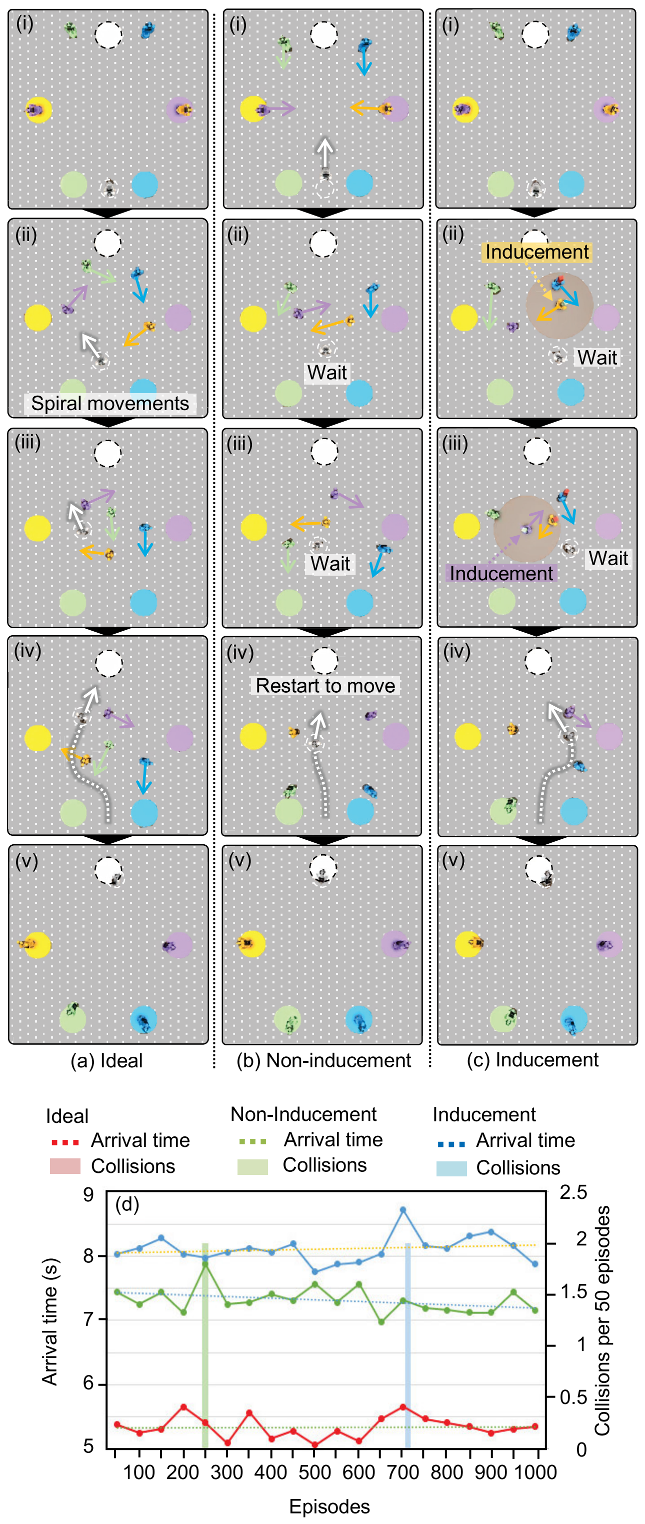


FIGURE 5. Results in S1 (open). (a) Ideal: perfect collaborative movement. (b) Non-inducement: standard movement of agents who have not been allowed to learn tendencies of other agents and are incapable of explicit actions. (c) Inducement: movement of agents who are capable of voice reminder. (d) Arrival time and number of collisions.

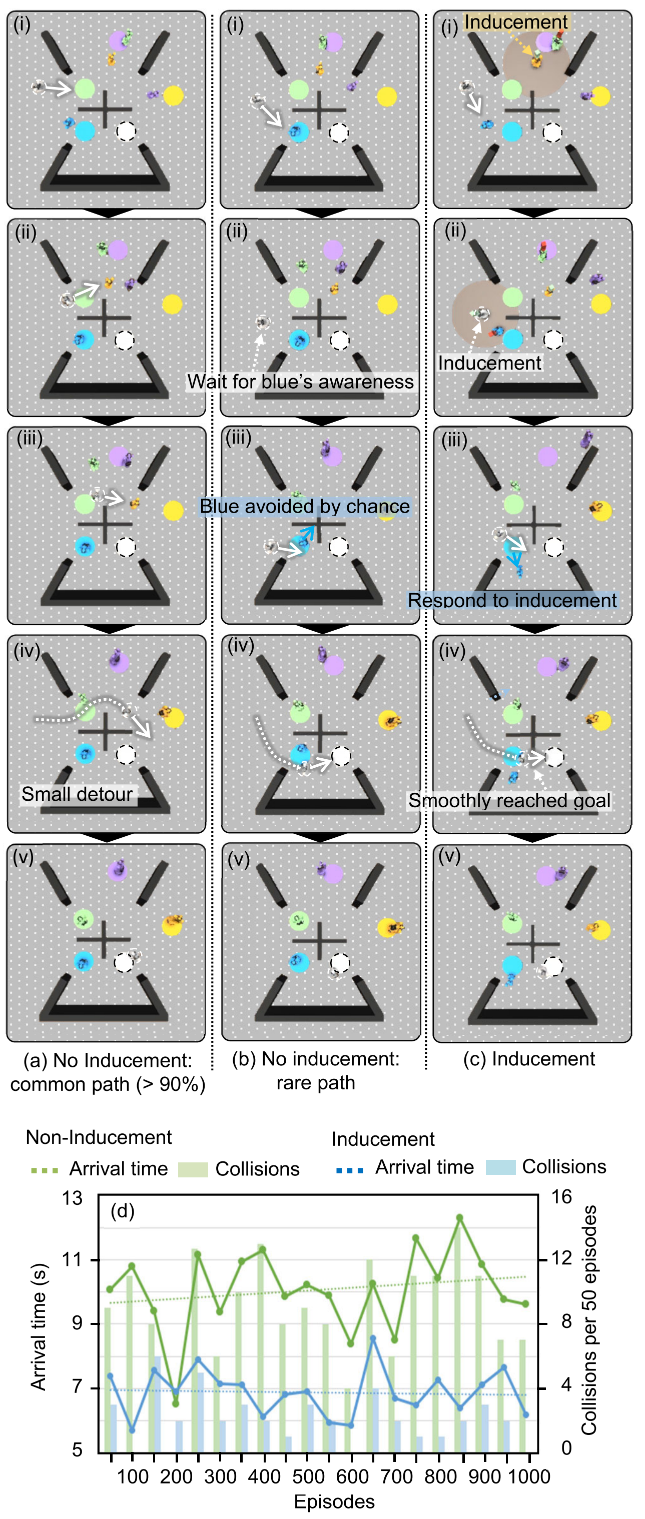


FIGURE 6. Results in S2-1 (small detours). (a) Non-inducement (common path (>90%)): white agent takes upper path and avoids agents to get to its goal. (b) Non-inducement (rare path): white agent waits until mutual awareness with blue agent then proceeds accordingly. (c) Inducement: white agent uses voice reminder to avoid waiting and proceeds accordingly. (d) Arrival time and number of collisions.

In the first route, which was most selected (>90%), as shown in Fig. 6 (a), the target agent took the upper path and passed by the other three agents before reaching its goal. While in Fig. 6 (b), in rare cases, the agent decided to wait for the blue agent's

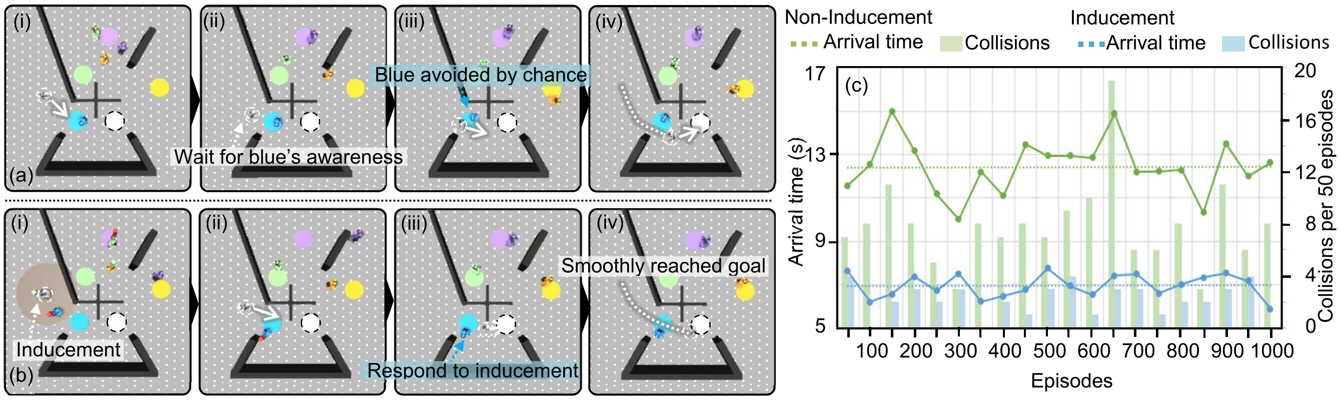


FIGURE 7. **Results in S2-2 (large detours). (a) Non-inducement: white agent decides that waiting on mutual awareness is better than taking the bottom path. (b) Inducement: white agent uses voice reminder to avoid waiting and reached goal smoothly. (c) Arrival time and number of collisions.**

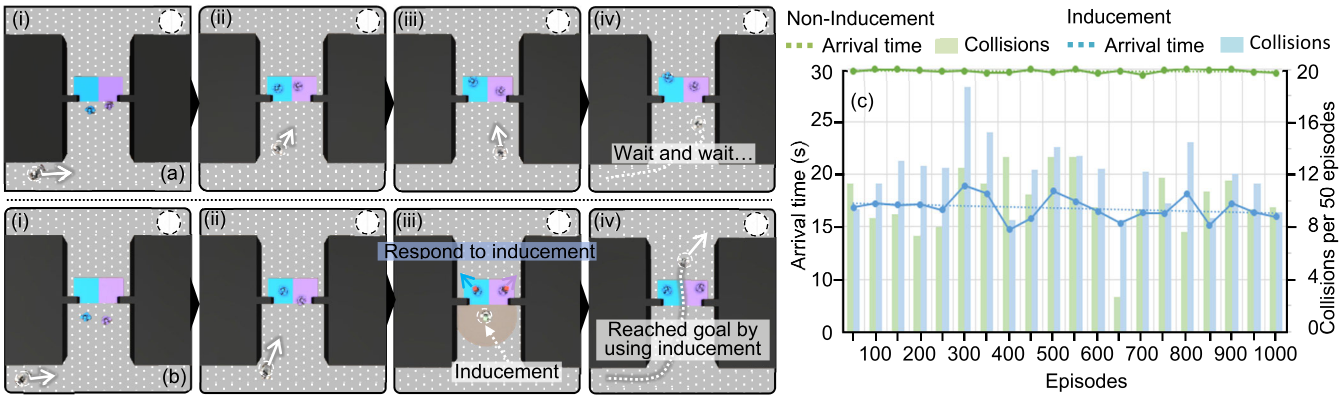


FIGURE 8. **Results in S3 (convergence). (a) Non-inducement: blue and purple agents cannot learn importance of letting white agent pass naturally when there is only one path. (b) Inducement: white agent directly conveys its needs, makes enough space to pass through, and arrives at goal. (c) Arrival time and number of collisions.**

awareness and take the shorter route. In inducement, Fig. 6 (c) shows that the target agent always took the center route and induced the blue agent once it was in range. The results in Fig. 6 (d) show that inducement was 37% more efficient than non-inducement. In addition, agents with a voice reminder (3.16) had less than half the average collisions compared with non-inducement (8.08). The primary reason for collisions for non-inducement was that four out of the five agents were consistently interacting within a closed space, which led to agents backing up into walls due to their limited field of vision. Overall, the success rates for non-inducement and inducement were 83.5 and 92.2%, respectively.

The main outcome of the experiment was that in cases where there are small detours, the use of inducement could be beneficial but is highly dependent on the degree of the detour.

C. EFFICIENCY ENVIRONMENT–LARGE DETOUR (S2-2)

We ran another test with the upper path closed off to see how agents would react. We found that both policies defaulted to taking the center route every time. In non-inducement, the agents *never* explored the inefficient route located at the bottom of Fig. 7 (a), rather they decided that the best course of action was to 'wait' on the blue agent's awareness or try to squeeze by. For inducement, as we expected, Fig. 7 (b) shows that agents took the same route as the one in S2-1 as the target agent never has to wait for awareness. By closing the top path, the arrival time for non-inducement increased by 126.2% to that of inducement. In inducement, the agents arrived at 5.5 s on average, while in non-inducement, it took 12.4 s, as shown in Fig. 7 (c). The collisions were similar to S2-1, but the success rate for the non-inducement (92.2%) was much closer to that of inducement (92.7%) at the expense of efficiency. This increase in success rate is due to the reduction of agents who pass through the highly congested upper path.

In scenarios with few paths and large detours, the integration of inducement had a much more noticeable effect on the average convergence time of agents.

D. CONVERGENCE ENVIRONMENT (S3)

This scenario was designed to mimic a situation with two unaware people talking in front of the only passage, but the agents were able to detect the target agent. The mutual awareness alone was insufficient for the model to converge to a policy where all agents could regularly reach their goals. In most episodes for non-inducement, the blue and purple agents would block the path, unaware of the target agent's desire, as shown in Fig. 8 (a). For inducement, Fig. 8 (b) shows that the agent could explicitly interact with the other agents to create a space sufficient to pass. Fig. 8 (c) shows that with a voice reminder, the agents could collectively reach their goal in 17.4 s. The agents with non-inducement were seldom able to reach all three goals simultaneously. A 30-s arrival time indicates that the agents could not lower their arrival time and the episode timed out. The average collisions were higher for inducement (13.2) than those for non-inducement (10.2). As this environment was the most restrictive, agents had little space to move when being induced. The main factor for failed convergence and the larger number of collisions was when induced agents would back up into the wall. For the success rates, non-inducement was 2.45% (there were extremely rare cases where the agent squeezed through the gap), and that for inducement was 66.2%. The major reason for lowering the success rate is that one of the two induced agents crashed during inducement. An approach to solve this issue would be to model the action of people looking around when the agent is induced.

E. DISCUSSION

1) INDUCEMENT USAGE

Along with evaluating the effectiveness of voice reminders in various settings, we also measured how often the voice reminder was used throughout all four environments. Fig. 9 shows the number of voice reminder usages per episode for the target agent in all the environments. The figure shows that inducement was used the least in the open environment (S1), used more as alternative paths were limited, and used the most in the convergence environment (S3). Note that each agent can use inducement up to three times per episode. We confirmed from the result that the effectiveness of inducement differs in environments.

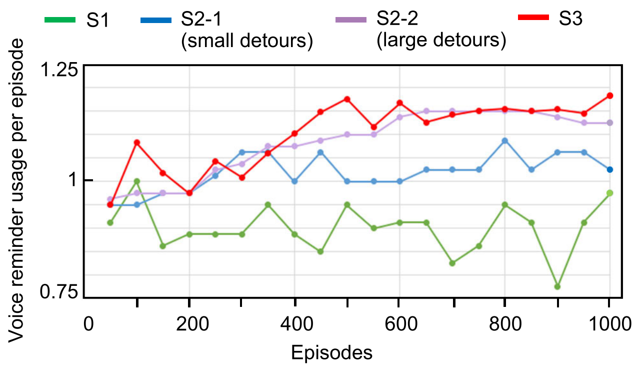


FIGURE 9. Voice reminder usage per episode for four different situations. The number of voice reminder usages is larger in order of S3, S2-2, S2-1, and S1.

2) CONTRIBUTION AND LIMITATIONS

We here summarize the contribution as well as limitations. This study is a preliminary trial to acquire an inducing policy using a multiagent deep reinforcement framework. The contribution is to obtain an interpretation of the effectiveness and necessity of using inducement, which can be a large forward step for updating interactive navigation. This study proved that inducement is effective in the time efficiency of a mobile robot. It also shows that the agents can learn the importance of inducements in restrictive environments over open ones. This is possible because the multiagent reinforcement learning framework clarifies when the agents use inducement and when they accept it. The results can be applied to parameter settings for rule-based navigation methods [10]. Meanwhile, the model we used is limited in how it rewards inducement responses. Agents are rewarded and punished for reacting and not reacting to inducement, respectively. In some situations, reacting to inducement is not suitable for an agent, so we need to define and incentivize a proper reaction. For future improvement, we will focus on heterogeneous agents since it would enable variable responses and modeling different types of agents that would reinforce the need for other inducements and responses.

V. CONCLUSION

To understand and evaluate if inducement (voice reminder) is effective and how and when it must be used, we propose to comprehend them through multiagent deep reinforcement learning (MDRL) in which the robot voluntarily acquires an inducing policy suitable for the situation. We introduced an inducement-based multiagent path planning algorithm that enables agents to use voice reminders to help convey their desires while traversing to their goals. We developed four different situations to evaluate how agents would implicitly and explicitly interact with one another. Experimental results validated our preconceptions that inducement can greatly improve group efficiency in situations where agents are unaware of another agent's presence. Inducement in open environments could be harmful to the group's efficiency. Meanwhile, inducement increases agent efficiency and detour severity in environments with more severe detours. Finally, in a single blocked passage, inducement was the determining factor for group convergence. The results also conveyed that the agents were more prone to using voice reminders when there was a lack of open space.

In future works, we plan on extending the types of explicit inducement actions by adding physical touch, enriching input information such as attention to collision risk [32], and integrating this framework into physical multi-robot systems.

ACKNOWLEDGMENT

The authors would like to thank the Waseda Research Institute for Science and Engineering Institute and the Future Robotics Organization of Waseda University.

REFERENCES

[1] W. Yuan, Z. Li, and C.-Y. Su, "Multisensor-based navigation and control of a mobile service robot," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 4, pp. 2624–2634, Apr. 2021.

[2] S. N. J. Pooganam, B. Gopalakrishnan, V. S. S. B. K. Avula, A. K. Singh, K. M. Krishna, and D. Manocha, "Reactive navigation under non-parametric uncertainty through Hilbert space embedding of probabilistic velocity obstacles," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2690–2697, Apr. 2020.

[3] X.-T. Truong and T. D. Ngo, "Toward socially aware robot navigation in dynamic and crowded environments: A proactive social motion model," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 4, pp. 1743–1760, Oct. 2017.

[4] Y. Chen, C. Liu, B. E. Shi, and M. Liu, "Robot navigation in crowds by graph convolutional networks with attention learned from human gaze," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2754–2761, Apr. 2020.

[5] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 797–803.

[6] M. Kamezaki, A. Kobayashi, Y. Yokoyama, H. Yanagawa, M. Shrestha, and S. Sugano, "A preliminary study of interactive navigation framework with situation-adaptive multimodal inducement: Pass-by scenario," *Int. J. Social Robot.*, vol. 12, no. 2, pp. 567–588, May 2020.

[7] M. C. Shrestha, T. Onishi, A. Kobayashi, M. Kamezaki, and S. Sugano, "Communicating directional intent in robot navigation using projection indicators," in *Proc. 27th IEEE Int. Symp. Robot Hum. Interact. Commun. (RO-MAN)*, Aug. 2018, pp. 746–751.

[8] M. C. Shrestha, Y. Tsuburaya, T. Onishi, A. Kobayashi, R. Kono, M. Kamezaki, and S. Sugano, "A preliminary study of a control framework for forearm contact during robot navigation," in *Proc. 27th IEEE Int. Symp. Robot Hum. Interact. Commun. (RO-MAN)*, Aug. 2018, pp. 410–415.

[9] M. Kamezaki, A. Kobayashi, R. Kono, M. Hirayama, and S. Sugano, "Dynamic waypoint navigation: Model-based adaptive trajectory planner for human-symbiotic mobile robots," *IEEE Access*, vol. 10, pp. 81546–81555, 2022.

[10] M. Kamezaki, Y. Tsuburaya, T. Kanada, M. Hirayama, and S. Sugano, "Reactive, proactive, and inducible proximal crowd robot navigation method based on inducible social force model," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3922–3929, Apr. 2022.

[11] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3826–3839, Sep. 2020.

[12] B. Brandão, T. Woerle De Lima, A. Soares, L. Melo, and M. R. O. A. Maximo, "Multiagent reinforcement learning for strategic decision making and control in robotic soccer through self-play," *IEEE Access*, vol. 10, pp. 72628–72642, 2022.

[13] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 6015–6022.

[14] A. Guillen-Perez and M.-D. Cano, "Learning from Oracle demonstrations—A new approach to develop autonomous intersection management control algorithms based on multiagent deep reinforcement learning," *IEEE Access*, vol. 10, pp. 53601–53613, 2022.

[15] J. N. Foerster, Y. M. Assael, N. De Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Proc. Conf. Neural Inf. Process. Syst.*, 2016, pp. 2145–2153.

[16] T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1–10.

[17] J. Wang, Z. Ren, T. Liu, Y. Yu, and C. Zhang, "QPLEX: Duplex dueling multi-agent Q-learning," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–27.

[18] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. Rabbat, and J. Pineau, "TarMAC: Targeted multi-agent communication," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1–10.

[19] W. Liu, S. Liu, J. Cao, Q. Wang, X. Lang, and Y. Liu, "Learning communication for cooperation in dynamic agent-number environment," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 4, pp. 1846–1857, Aug. 2021.

[20] F. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*. Cham, Switzerland: Springer, 2016.

[21] A. Juliani, V.-P. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A general platform for intelligent agents," 2018, *arXiv:1809.02627*.

[22] A. Cohen, E. Teng, V.-P. Berges, R.-P. Dong, H. Henry, M. Mattar, A. Zook, and S. Ganguly, "On the use and misuse of absorbing states in multi-agent reinforcement learning," 2021, *arXiv:2111.05992*.

[23] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 2, pp. 2974–2982.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[25] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, 1988.

[26] *Unity ML-Agents*. Accessed: Jan. 10, 2023. [Online]. Available: https://github.com/Unity-Technologies/ml-agents/blob/main/docs/Training-Configuration-File.md

[27] D. Ha, A. Dai, and Q. Le, "Hypernetworks," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–18.

[28] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 2778–2787.

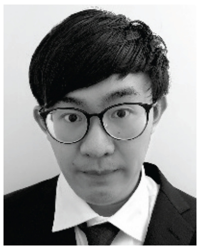
[29] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Conf. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6379–6390.

[30] R. Wang, M. Everett, and J. How, "R-MADDPG for partially observable environments and limited communication," in *Proc. ICML Workshop RL4RealLife*, 2019, pp. 1–9.

[31] *Unity ML-Agents*. Accessed: Jan. 10, 2023. [Online]. Available: https://github.com/Unity-Technologies/ml-agents/blob/main/docs/Learning-Environment-Design-Agents.md

[32] M. Hayashi, T. Miyake, M. Kamezaki, J. Yamato, K. Saito, T. Hamada, E. Sakurai, S. Sugano, and J. Ohya, "Preliminary investigation of collision risk assessment with vision for selecting targets paid attention to by mobile robot," in *Proc. 31st IEEE Int. Conf. Robot Hum. Interact. Commun. (RO-MAN)*, Sep. 2022, pp. 624–629.

MITSUHIRO KAMEZAKI (Member, IEEE) received the B.S., M.S., and Dr.Eng. degrees in mechanical engineering from Waseda University, Tokyo, Japan, in 2005, 2007, and 2010, respectively. From 2007 to 2008, he was a Visiting Research Associate with the 21st Century COE Program, Waseda University. From 2008 to 2010, he was a Research Associate with the Global COE Program, Waseda University. From 2010 to 2013, he was a Research Associate with the Department of Modern Mechanical Engineering, Waseda University, where he is currently an Associate Professor with the Research Institute for Science and Engineering. His current research interests include intelligent machine systems, man–machine interface, and operator support system in operated-work machines. He is a member of the IEEE RAS, SICE, RSJ, and JSME. He received the Best Paper Award from the 11th Symposium on Construction Robotics in Japan, in 2008, the IROS Best Paper Award Finalist from the IEEE/RSJ IROS, in 2011, the SICE Young Author's Award Finalist from the SICE Annual Conference, in 2013, the Young Investigation Excellence Award from RSJ, in 2016, the Best Paper Award from the IEEE/ASME AIM, in 2016, and the IEEE Robotics and Automation Letters Best Paper Award, in 2022.



RYAN ONG received the B.S. degree from the Department of Mechanical Engineering, University of Nevada, Las Vegas, USA, in 2018, and the M.S. degree in modern mechanical engineering from Waseda University, Tokyo, Japan, in 2022. His research interests include multi-agent systems, reinforcement learning, and autonomous mobile robots.

SHIGEKI SUGANO (Fellow, IEEE) received the B.S., M.S., and Dr.Eng. degrees in mechanical engineering from Waseda University, Tokyo, Japan, in 1981, 1983, and 1989, respectively. Since 1986, he has been a Faculty Member with the Department of Mechanical Engineering, Waseda University, where he is currently a Professor. Since 2014, he has been the Dean of the School/Graduate School of Creative Science and Engineering, Waseda University. His research interests include human-symbiotic anthropomorphic robot design, dexterous and safe manipulator design, and human–robot communication. He is a fellow of four academic societies, such as the Japan Society of Mechanical Engineers (JSME), the Society of Instrument and Control Engineers (SICE), and the Robotics Society of Japan (RSJ). He served as the General Chair for the IEEE/ASME International Conference on Advanced Intelligent Mechatronics, in 2003, and the IEEE/RSJ International Conference on Intelligent Robots and Systems, in 2013. From 2001 to 2010, he served as the President for the Japan Association for Automation Advancement. He was the General Co-Chair of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2006) and the Program Co-Chair of the 2009 IEEE International Conference on Robotics and Automation (ICRA2009). He also served as the General Co-Chair for the 2012 IEEE International Conference on Robotics and Automation (ICRA2012) and the Program Chair for the 2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM2012). In 2017, he served as the President for SICE.