# Learning Foresighted People Following under Occlusions

AbdElMoniem Bayoumi Philipp Karkowski Maren Bennewitz

***Abstract*— In many situations, users walk on typical paths between specific destinations at which the service of a mobile robot is needed. Depending on the environment and the paths, step-by-step following of the human might not be the optimal solution since better paths for the robot exist. We propose to perform a prediction about the human's future movements and use this information in a reinforcement learning framework to generate foresighted navigation actions for the robot. Since frequent occlusions of the human will occur due to obstacles and the robot's constrained field of view, the estimate about the humans's position and the prediction of the next destination are affected by uncertainty. Our approach deals with such situations by explicitly considering occlusions in the reward function such that the robot automatically considers to execute actions to get the human in its field of view. We show in simulated and real-world experiments that our technique leads to significantly shorter paths compared to an approach in which the robot always tries to closely follow the user and, additionally, can handle occlusions.**

## I. Introduction

In this work, we consider scenarios in which a service robot needs to encounter a user at designated locations, between which the human moves along commonly used paths. One example application are transportation tasks where the robot has to provide items to the human at predefined places. Following the human at a certain distance might enable the robot to solve this task and various techniques exist to closely follow people [1]–[6]. However, since humans may not always take the shortest path to their next destination these methods might lead to an inefficient robot navigation behavior in the considered scenarios where the robot has to encounter the human *only* at designated places. Furthermore, the user might occasionally move through passages that are impassable to the robot such that the robot is forced to find an alternative route. In such cases the problem arises that the human will eventually be out of the robot's field of view, which will lead to an uncertain estimate about the human's position and a wrongly predicted destination. Thus, the robot will need to consider active re-localization of the human to improve the estimate and to be able to infer the next navigation goal.

In this paper, we apply an approach to constantly predict the human's position at future time steps based on the robot's previous observations of the human's position. This prediction is used as an input to a reinforcement learning framework to generate foresighted robot navigation actions.

All authors are with the Humanoid Robots Lab, University of Bonn, Germany. This work has been supported by the German Academic Exchange Service (DAAD) and the Egyptian Ministry for Higher Education as well as by the European Commission under contract number FP7-610532-SQUIRREL and by the DFG-Project FOR 2535 Anticipating Human Behavior.

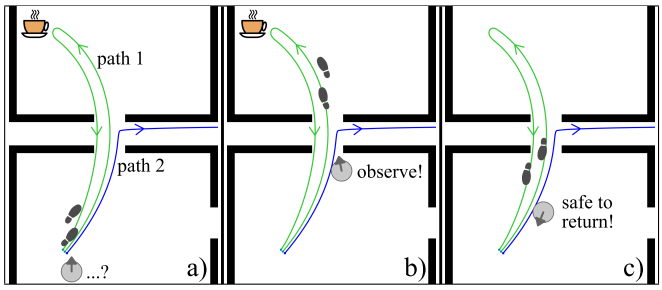


Fig. 1. Example of foresighted people following: In a scenario where a human commonly walks on different known paths, our approach enables the robot to act efficiently. a) When the human starts walking the robot does not yet know which of the two paths the human will take and simply follows. b) The robot predicts the path of the human and remains at a spot, where possible deviations from the prediction are well visible. c) The robot sees the human return and immediately drives back to the initial spot.

The resulting navigation strategy leads to efficient robot paths and observation actions that improve the prediction of future human movements.

Fig. 1 highlights the strengths of our approach. The human initially moves along a trajectory that might correspond to two different paths, while the robot is only needed if the human moved along path 2. Our approach leads to an efficient strategy where the robot follows the human to the place where the paths diverge and observes the future behavior of the human, hence, reducing unnecessary movement actions. Following the human at close distance would also be a solution to reduce uncertainty, but this often results in unnecessary detours and interferences with the human.

An overview of our proposed system is shown in Fig. 2. Initially, we use the motion prediction model to learn navigation strategies via reinforcement learning. The output is a Q-table that provides the robot with the best navigation action for the current situation.

As we show in simulated and real-world experiments, our framework generates foresighted navigation actions that can also deal with cases in which the human is not in the robot's field of view. The trajectories generated by our approach are significantly shorter compared to a direct-following approach and we achieve a higher number of successful runs. We extended our previous work [7] by using a more compact representation of the movement possibilities, implementing a particle filter based prediction, modifying the state space, enhancing the reward function to deal with occlusions, and allowing the robot to explicitly perform observation actions to update the belief about the human's location.

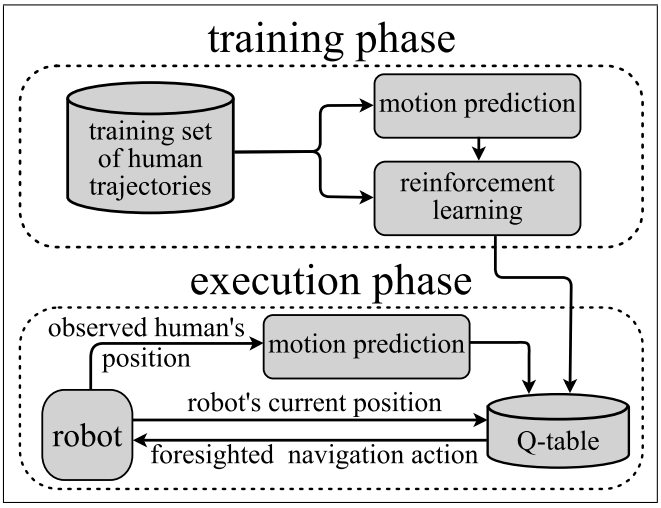


Fig. 2. Based on the robot's current position and its observations about the human, the robot chooses a navigation action using the foresighted navigation strategy that was learned off-line from a training set of observed human trajectories.

## II. Related Work

Various related approaches have been presented. For example, Tipaldi and Arras [8] learned spatio-temporal models of human activities in order to increase the probability for the robot to encounter specific humans. Krajník *et al.* [9] presented a method based on spatio-temporal models to help the robot finding non-stationary objects in an office environment. The authors represent the environment as a topological map and combine it with periodic functions in order to compute the likelihood of existence of these objects at any node of the map with respect to the time. The focus of these works is different than ours since they aim at quickly finding a specific person/object in the environment whereas we aim at foresighted, efficient people following in both cases whether the human is in the field of view or not.

Goldhoorn *et al.* [10] introduced a two-stage approach to find a moving person first and then following them. The authors presented continuous real-time partially observable Monte-Carlo planning (CR-POMCP) to generate a policy tree through simulations. Afterwards, the authors extended the CR-POMCP by combining it with a standard direct following approach. Thus, the robot moves directly to the human as long as it is in the field of view. If the human is not visible, the authors use the human position with the highest probability according to the CR-POMCP as the next navigation goal of the robot. In contrast, our approach reasons about the next destination of the human and uses the learned policy to decide on the best action, thereby explicitly taking into account observation actions. Our system does the reasoning in both cases whether the human is in the field of view or not to generate efficient robot behavior.

Kretzschmar *et al.* [11] learned a probabilistic model of pedestrians from observing their trajectories. Kuderer and Burgard [1] applied that approach to predict a human's trajectory and plan an intelligent path for a robot following the human. The proposed method takes into account information about obstacles in the environment and computes the robot's trajectory so that the distance to the desired relative position along the predicted human path is minimized. In our work, we go beyond such an idea and take into account long-term prediction of the humans motions and compute efficient robot actions, instead of staying always in the vicinity of the human.

The contribution of our work is the generation of foresighted robot behavior based on learned navigation strategies and the prediction of human motion. The prediction of human motions itself is not the focus. Nevertheless, we present some recent work in this area in the following. Best and Fitch [12] presented a Bayesian trajectory prediction model for moving people. The authors assume that there is a set of predefined destinations between which the person moves on the shortest path and, therefore, the likelihood of a movement is inversely proportional to how far is the corresponding future position from the shortest path to a given destination. Ziebart *et al.* [13] introduced an approach based on a softened Markov decision process (MDP) that is trained on a set of observed human trajectories and uses a Q-table to encode the human's movements in a grid map at any point at any time. The authors proposed to make use of this modeling for planning collision-free paths. Vasquez [14] extended the work of Ziebart *et al.* [13] in order to decrease the computational complexity. The authors here assume that the cost function based on which the human plans its motion is given a priori. We previously implemented the technique of Ziebart *et al.* [13] but noticed in our experiments that we could not achieve good prediction results in case the human deviates from learned trajectories. Therefore, for this work we apply a particle filter based motion prediction.

In contrast to our previous work [7], the new framework relies only on the robot's on-board sensors and can deal with occlusions. Moreover, with the modified state space we are able to handle also large environments without overloading the learning process.

## III. Problem Formulation

We consider a scenario where the robot knows the structure and static obstacles in the environment. We assume that the human commonly walks on typical paths between different points of interest (destinations) where a robot is needed to assist the human. Our aim here is to exploit the given information in order to generate more efficient navigation actions for the robot compared to a direct-following method and additionally deal with situations where such a method would fail. An example of the former is discussed in Fig. 1. Furthermore, since it sometimes happens that the human moves along paths which are impassable for the robot, the robot should learn in these cases how to move to the predicted destination of the human, thereby performing explicit observation actions to update the prediction of the human position if useful. All in all, the aim of our approach is to reach the correct destination of the human while reducing the overall path length.

## IV. Motion Prediction

To track the user while moving along paths between a set of predefined destinations, we use a particle filter following ideas of the work by Liao *et al.* [15]. The particles are propagated, over the nodes of a topo-metric graph that represents the environment (see Fig. 3), according to the user's velocity. The poses of the particles are initialized at the graph node that corresponds to the user's known starting location. Each particle moves on the graph toward one of the destinations, reachable from the previous destination (the user's starting location). In particular, at every time step, each particle either moves to the neighboring node along the shortest path to its intended destination, stays at the current node, or moves to another neighboring node. The choice between these alternatives follows a certain probability distribution that we determined experimentally.

The weights of the particles correspond to the observation likelihood. High weights are assigned to the particles that are close to the user's observed position. In case the user is not observed, the weights of the particles in the field of view are reduced.

To predict the user's future location, we simulate the propagation of the particles according to the motion model described above a few time steps into the future and then determine the predicted graph node based on the cluster of particles that has the highest weight.

## V. Learning Navigation Actions

In order to generate actions for the robot while observing and following the human, we use reinforcement learning and combine it with the human motion prediction (see Section IV). For every possible starting location of the human (corresponding to the points of interest/destinations), we learn a separate Q-table that represents the navigation policy of the robot. The remainder of this section explains in detail all aspects of our learning approach.

### A. Reinforcement Learning

In reinforcement learning, the state of the robot $s_t$ from the state space $\mathcal{S}$ is transformed to $s_{t+1}$ by executing action $a_t$ from the action space $\mathcal{A}$ at time $t$. This transformation takes place according to the transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ and the robot gets an immediate reward $r_t \in \mathcal{R}$ based on the reward function.

The learning process is iterative, where each iteration is called a learning episode. $Q^\pi(s_t, a_t)$ refers to the expected reward that can be achieved if action $a_t$ is executed at state $s_t$. In each learning episode, the Q-value of each state-action pair belonging to this episode is updated based on the obtained reward $R_t$ following the basic Sarsa rule:

$$R_t = \sum_{i=t+1}^{T} r_i \tag{1}$$

$$Q^\pi(s_t, a_t) = E_\pi\{R_t | s_t, a_t\} \tag{2}$$

$$Q^\pi(s_t, a_t) \leftarrow Q^\pi(s_t, a_t) + \alpha[r_{t+1} + \gamma Q^\pi(s_{t+1}, a_{t+1}) - Q^\pi(s_t, a_t)], \tag{3}$$

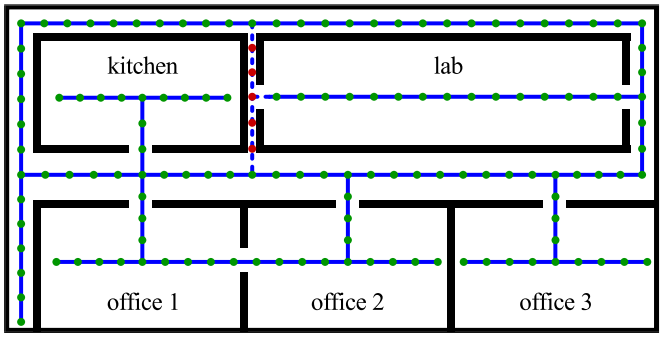


Fig. 3. Simulation environment with topo-metric graph. The environment is represented as a grid map with an overlaid topo-metric graph, such that each grid cell corresponds to the nearest graph node (green dots) within the same room. At the same time, each graph node is mapped onto the grid cell, which lies at the same position. The red dots correspond to nodes that are only passable by the human but not by the robot.

where $\alpha$ and $\gamma$ refer to the step size and discounting factor, respectively. Instead of Sarsa, which only considers the next time step, we used the advanced Sarsa($\lambda$) that considers several steps forward when updating the Q-value, which leads to a more accurate estimate of $R_t$ [16].

The action policy $\pi$ should aim at maximizing the return of each state, which corresponds to choosing the action with the maximum Q-value. However, since the robot may get stuck in a local-minima at a certain graph node, we apply $\varepsilon$-greedy action selection to allow more exploration of the state-action search space. This policy selects the action with the maximum Q-value with probability $1 - \varepsilon$ or any of the other non-greedy actions randomly with probability $\varepsilon$.

### B. State Space $\mathcal{S}$

As mentioned above, for the prediction of the human's motion, we represent the environment as a discretized grid map with an overlaid topo-metric graph. For the state space and the action generation we use the same representation, such that every grid cell is mapped onto the closest graph node within the same room. Additionally, every graph node is mapped onto the grid cell, which lies at the same position as the node. The topo-metric representation is a compact representation of the movement possibilities in the environment and better scales with larger maps. The corresponding map representation as topo-metric graph for our quantitative simulation experiments is depicted in Fig. 3. Note that the environment contains a passage that is not passable by the robot (red nodes) and, thus, the corresponding navigation actions are excluded.

The state space includes the current graph node position of the robot $\mathbf{x}_t^r$ at time $t$ and the predicted position of the human, also mapped onto the graph, $\mathbf{x}_{t+i}^h$ after $i$ future time steps:

$$s_t = \begin{bmatrix} \mathbf{x}_t^r \\ \mathbf{x}_{t+i}^h \end{bmatrix}. \tag{4}$$

This state space is much more concise compared to our previous work [7] and leads to a faster learning process.

We do not include the human's current position here since it does not give more information and, apart from that, might not be observable. In contrast to our previous state representation, the new formulation can scale for reasonably large environments.

The topo-metric graph does naturally not cover the whole grid map and the real robot might not perfectly drive along the graph. Therefore, the graph node position of the robot is always determined according to the robot's current grid cell mapped onto the graph, similarly as for the human.

### C. Action Set $\mathcal{A}$ and Q-tables

The action space consists of movement actions, a waiting action, and an observation action. Movement actions are defined along the graph and consist of movements to all neighboring nodes. Therefore, the amount of movement actions depends on the current node position of the robot, as certain nodes have more neighbors compared to others, e.g., intersections. An observation action consists of a full 360° rotation or a rotation until the human is observed.

In order to execute actions in the real world we need to map the action space, i.e., movements between neighboring graph nodes, onto real actions of the robot. We accomplish this by determining the corresponding grid cell position of the graph node after performing an action and sending the real 2D coordinate as a new local goal to the robot.

### D. Reward Function $\mathcal{R}$

During the training phase, we define the immediate reward $r_t$ at time $t$ as follows:

$$r_t = \begin{cases} 10,000 & \text{if } t = T \\ -A^*(\mathbf{x}_t^r, \mathbf{x}_D^h) - v \cdot A^*(\mathbf{x}_t^r, \mathbf{x}_{t+i}^h) - C & \text{otherwise} \end{cases}, \tag{5}$$

where $T$ is the final time step in the case of a successful learning episode. A learning episode is considered as successful if the robot reaches the human's destination through a shorter path compared to an approach that uses the concept of directly following, which also moves along the graph, however, at every time step simply moves one node toward the human's current position.

The function $A^*(\mathbf{x}_t^r, \mathbf{x}_D^h)$ in Eq. (5) denotes the distance resulting from applying the A* algorithm to get the shortest path from the robot's current position $\mathbf{x}_t^r$ to the human's intended destination $\mathbf{x}_D^h$, which is known from the training data. The term $A^*(\mathbf{x}_t^r, \mathbf{x}_{t+i}^h)$ represents the A* distance between the robot's current position and the position of the human after $i$ time steps, while $v$ is 0 in case the human is currently visible and 1 otherwise. Finally, $C$ represents the cost for performing an action, which is 0 in case of a waiting and observation action and the distance between two neighboring nodes in case of a movement action.

The first term of Eq. (5) guides the robot toward the goal while the second term helps to keep the human in the robot's field of view. The last term minimizes the total amount of movement actions, which is desirable if the human takes detours or moves to areas where the robot is not required.

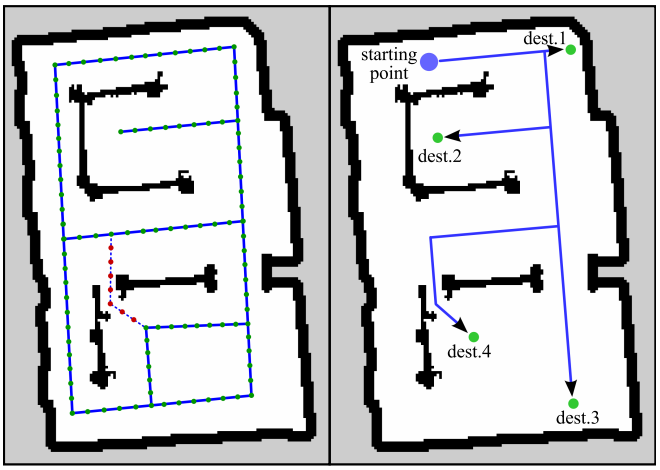


Fig. 4. Left: The real-world environment with an overlaid topo-metric graph. The red dots correspond to nodes that are only passable by the human but not by the robot. Right: Common human trajectories used in the real-world environment.

## VI. Experiments

### A. Experimental Setup

We evaluated our approach, both in simulated and real-world experiments. The size of the simulation environment is $41\,\text{m} \times 20.5\,\text{m}$ with a map resolution of $0.25\,\text{m}$ and a map of size $4.8\,\text{m} \times 7.6\,\text{m}$ with a resolution of $0.05\,\text{m}$ for the real-world experiment (see Fig. 3 and Fig. 4, respectively). The distances between the graph nodes in the simulated and real-world environments are $1.5\,\text{m}$ and $0.5\,\text{m}$, respectively. To show one strength of our approach, each of the environments contains a narrow passage that is not passable by the robot. Thus, directly following the human along these passages is not possible and will not succeed, while our approach tries to predict the destination of the human and learns how to best move in order to encounter the human at the destination, thereby considering also observation actions along the way. Note that the heuristic strategy of just moving to the predicted destination on the shortest path does not consider observation actions and will sometimes be affected by wrong predictions.

In case the human is outside the robot's field of view, we propagate the particles and update their weights, i.e., reduce their weights if they are within the robot's field of view, and then determine the predicted graph node based on the cluster of particles that has the highest weight.

During the reinforcement learning phase, we initialize $\varepsilon$ of the greedy action selection policy with $0.35$ to allow for a wider exploration of the state and action space. During the execution phase, the robot follows the learned policy represented by the Q-table, i.e, $\varepsilon = 0$. Each Q-table is learned in simulation from a maximum of $4000$ learning episodes with a maximum of $100$ time steps. We experimentally determined that a three-step prediction ahead, i.e., $i = 3$ in Eq. (4) works best.

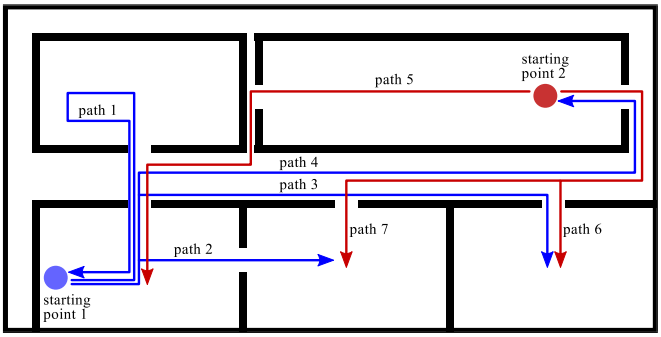


Fig. 5. Common human trajectories used in the simulation environment. Note that path 1 in this figure corresponds to the example shown in Fig. 1.

Fig. 6. The robot and the user with a board of ArUco markers to simplify pose estimation in the real-world experiment.

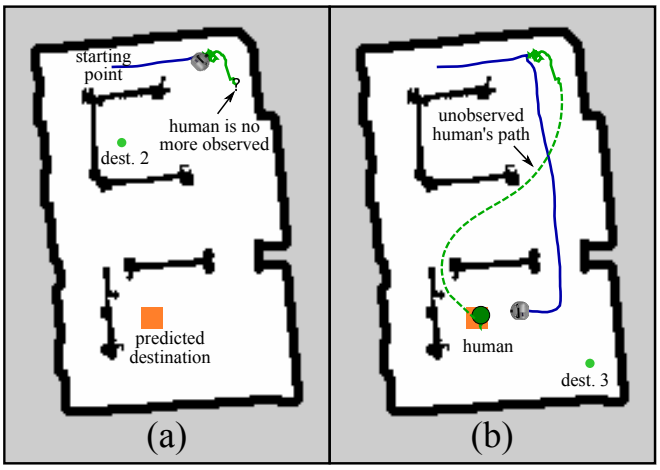


Fig. 7. Real-robot experiment: (a) The human gets outside the robot's camera field of view while walking along his path (green trajectory) toward his destination. The robot keeps updating the prediction until it reaches the correct destination. (b) The robot executes the learned actions to reach the destination successfully through an alternative path (blue trajectory) that avoids the narrow entrance that does not fit the robot size.

### B. *Experiments in Simulation*

For the simulation experiments we simulated 105 human trajectories, 15 for each possible route (see Fig. 5). 60% of the trajectories were used for the training and the rest for the evaluation of the learned robot behavior.

As a performance measure, we computed the reduction in path length to evaluate the efficiency of the generated motion actions, e.g., to assess battery saving. We compare the path resulting from applying our framework to the path generated from the direct-following strategy, in which the robot moves at each time step toward the node of the human's position. This presents the best heuristic behavior the robot can execute as the actual destination can only be predicted and might be wrong. We evaluated the statistical significance using a *two-tailed paired z-test*.

The experimental results show that our learned policy significantly outperforms the direct-following approach in terms of path length (statistical significance of 95%), which was reduced by 7.33%. Additionally, the percentage of runs in which the robot failed to reach the final destination of the user, is reduced to only 1% compared to 14% in the case of the direct-following approach.

### C. *Real-World Experiment*

In addition to the experiments in simulation, we performed an experiment with a real robot. We used a Robotino robot from Festo and used the on-board laser sensor for particle filter based localization on the given grid map. To estimate the human's position, the user holds a board of ArUco markers [17] as shown in Fig. 6. We focused on showing the ability of our framework to handle situations in which the human's trajectory also contains passages that are impassable for the robot so that simply following the human at close distance will fail.

The environment of the real-world experiment with the topo-metric graph is shown in the left image of Fig. 4. The Q-table of the learned navigation actions for this environment was also learned in simulation.

For the experiment shown in Fig. 7, the human moved toward his intended destination (which is unknown for the robot) through a path that involved passing through two narrow walls, which does not fit the size of the Robotino robot. Moreover, the human left the robot's field of view (due to the robot's orientation). The prediction, however, kept being updated until it reached a final destination. Meanwhile, the robot executed the learned Q-table actions based on the current robot position and the current prediction at each time step. As can be seen, the robot could successfully encounter the human at the intended destination. The robot took an efficient path that fits the robot size and at the same time considers potential prediction errors, i.e., it executed an observation action by rotating 360° that put destination 2 into its field of view (see Fig. 7 (a)). After that, the learned actions led to a path that put destination 3 in the field of view of the robot as it moved on its path (see Fig. 7 (b)). Therefore, if the prediction about the human pose was wrong and the human was waiting in any of these destinations, there would be a high probability of observing the human again and, thus, the prediction would have been updated accordingly. Note that in this scenario, if the robot navigates according to the strategy

of closely following the human it would get stuck in front of the narrow passage.

## VII. Conclusions

In this paper, we presented a reinforcement learning approach to generate efficient navigation actions for a service robot that needs to encounter a user at designated locations, between which a human moves on typical paths. We hereby apply a particle filter based prediction about the human's motion to deal with possible occlusions and generate effective navigation and observation actions.

In simulated as well as in real-world experiments, we showed that our method enables the robot to navigate efficiently to predicted destinations and also deal with cases in which the human is not within the robot's sensor range. In these situations, the robot explicitly considers to execute observation actions to re-observe the human and update the prediction about the destination. Additionally, the experimental results demonstrate that our framework generates significantly shorter paths and performs with a higher percentage of successful runs compared to a baseline approach in which the robot tries to follow the human at close distance.

## References


[1] M. Kuderer and W. Burgard, "An approach to socially compliant leader following for mobile robots," in *Social Robotics*, ser. Lecture Notes in Computer Science, M. Beetz, B. Johnston, and M.-A. Williams, Eds., vol. 8755. Springer International Publishing, 2014.

[2] T. P. Nascimento, A. P. Moreira, and A. G. Scolari Conceio, "Multi-robot nonlinear model predictive formation control: Moving target and target absence," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1502–1515, 2013.

[3] N. Pradhan, T. Burg, S. Birchfield, and U. Hasirci, "Indoor navigation for mobile robots using predictive fields," in *Proc. of the American Control Conference*, 2013.

[4] L. Huang, "Control approach for tracking a moving target by a wheeled mobile robot with limited velocities," *Control Theory & Applications, IET*, vol. 3, no. 12, pp. 1565–1577, 2009.

[5] I. Harmati and K. Skrzypczyk, "Robot team coordination for target tracking using fuzzy logic controller in game theoretic framework," *Robotics and Autonomous Systems*, vol. 57, no. 1, pp. 75–86, 2009.

[6] J. Huang, S. Farritor, A. Qadi, and S. Goddard, "Localization and follow-the-leader control of a heterogeneous group of mobile robots," *IEEE/ASME Trans. on Mechatronics*, vol. 11, no. 2, 2006.

[7] A. Bayoumi and M. Bennewitz, "Learning optimal navigation actions for foresighted robot behavior during assistance tasks," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2016.

[8] G. Tipaldi and K. Arras, "I want my coffee hot! Learning to find people under spatio-temporal constraints," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.

[9] T. Krajník, M. Kulich, L. Mudrová, R. Ambrus, and T. Duckett, "Where's waldo at time t? Using spatio-temporal models for mobile robot search," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2015.

[10] A. Goldhoorn, A. Garrell, R. Alquézar, and A. Sanfeliu, "Continuous real time POMCP to find-and-follow people by a humanoid service robot," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2014.

[11] H. Kretzschmar, M. Kuderer, and W. Burgard, "Learning to predict trajectories of cooperatively navigating agents," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2014.

[12] G. Best and R. Fitch, "Bayesian intention inference for trajectory prediction with an unknown goal destination," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[13] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa, "Planning-based prediction for pedestrians," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009.

[14] D. Vasquez, "Novel planning-based algorithms for human motion prediction," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2016.

[15] L. Liao, D. Fox, J. Hightower, H. Kautz, and D. Schulz, "Voronoi tracking: Location estimation using sparse and noisy sensor data," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2003.

[16] R. S. Sutton and A. G. Barto, *Introduction to reinforcement learning*. MIT Press, 1998.

[17] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and M. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280 – 2292, 2014.