Efficient Human Following Using Reinforcement Learning

AbdElMoniem Bayoumi Maren Bennewitz

***Abstract*— In this paper, we present an approach that relies on machine learning techniques to follow people efficiently during robotic assistance tasks, in which the robot is mainly interested in reaching the final navigation goal of the human. People can perform unexpected actions during navigation, which can lead to inefficient trajectories to the target destination (ex: answer land-line phones ... etc). Therefore, the following robot should infer the human's intended navigation goal and intelligently plan its own path to reach it, instead of just following the human's path. We propose a novel learning framework to generate such an efficient navigation strategy for the robot. In particular, we apply reinforcement learning from which we get a Q-function that computes for each pair of robot and human positions the best navigation action for the robot. Our approach applies a prediction of the human's motion based on a softened Markov decision process (MDP). This MDP is independent from the navigation learning framework and is learned beforehand based on previously observed trajectories. We thoroughly evaluated our approach in simulation and on a real robot. As the experimental results demonstrate, our approach leads to an efficient navigation behavior during the following task and can significantly reduce the path length and completion time compared to naive following strategies.**

I. Introduction

Following humans with mobile robots is a challenging problem and is needed in several applications such as in industrial settings where robots are deployed as transportation systems, in home scenarios, especially for the elderly people, in stores with autonomous shopping carts, or in environments where robotic wheelchairs should navigate next to an accompanying pedestrian. Various solutions have been proposed for such instances of the robotic following task as discussed in the next section. However, these solutions mainly focus on following the human in a certain distance irrespective of its intended final destination. In other words, the robot will keep on following the human, even if it moves on an inefficient path to that destination. This inefficiency may be due to the fact that the human can be easily interrupted by an unexpected event that may arise. For example, consider the case of a home-assistance robot performing a transportation task and following human when suddenly a land-line phone rings, or the doorbell rings in the middle of the task. In such cases, a robot applying naive following strategies can only follow the human since the destination is unknown to the robot. This leads to inefficient navigation behavior causing unnecessary battery consumption or wear.

All authors are with the Institute of Computer Science, 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.

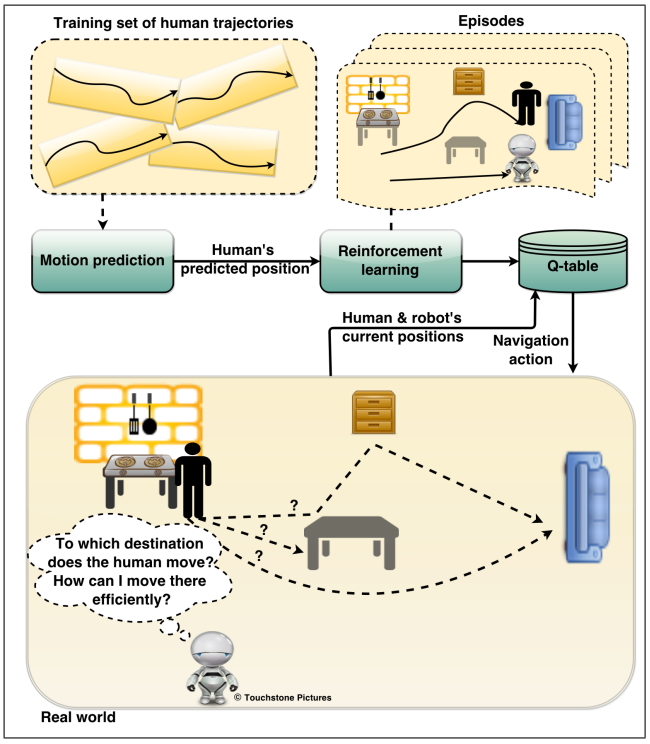


Fig. 1. The human moves through the environment between different destinations (bottom image) where it stays for a while and may need the help of a mobile robotic assistant. The task of the robot is to reach the initially unknown destination of the human in an efficient and timely manner (top right image). We developed a learning framework that generates efficient navigation actions based on predicted motions of the human.

We developed a reinforcement learning framework that leads to efficient navigation behavior in the described situations and can handle such unexpected human behaviors. Our approach continuously predicts the human's intended destination and based on that the robot moves so as to reach the goal efficiently while following a learned navigation strategy. The prediction of the human path is updated every time step and, thus, the robot is able to adapt its own actions if necessary.

An overview of our framework is depicted in Fig. 1. We rely on a set of previously observed human trajectories between possible destinations where the human stays for a while and might need the help of the robot, i.e., for general assistance tasks, social interaction, or delivery tasks. Given these trajectories, our approach learns a prediction model that can be used to reason about the future motions and target destination of the human given its trajectory observed so far. The output of our learning method is a table of Q-values that encodes the most efficient navigation action for the robot

considering the current robot and human position as well as the predicted human motion.

As we show in the experiments, our approach leads to an efficient navigation behavior. The robot focuses on reaching the human's final destination – which is initially unknown to the robot – regardless of the inefficiency of the human target's trajectory. We demonstrate that a robot executing the learned navigation strategy can successfully handle the cases that the human trajectory contains detours or unexpected behaviors. Our method results in efficient paths with a significantly reduced completion time compared to naive following strategies. To the best of our knowledge, we present the first solution to a robotic following application that considers the efficiency of the generated robot paths.

II. Related Work

Robotic following tasks have been thoroughly investigated using control theory, e.g., Huang [1] presented two control models for both the linear and angular velocity of a wheeled robot. These control models focus on originating velocity commands that allow the robot to track the target while ensuring smoothness of the resulting trajectory. Nascimentoa *et al.* [2] developed an approach for cooperative target tracking and proposed a nonlinear predictive formation control model for the robots in a distributed architecture. Each robot shares information about its pose with the rest of the robots team and optimizes its control signals under a prediction of the next state of the target as well as the other team members' states. Pradhan *et al.* [3] proposed a path planning method with a navigation function that uses predictive fields of moving obstacles to follow a target. Choi *et al.* [4] presented a model for the target-following task in an environment equipped with RFID tags that are used by the robots for localization. The velocity of the following robot is computed according to the leading robot's relative position, it is inversely proportional to the relative distance.

Furthermore, there are approaches that aim at following a person at a fixed distance, e.g., Prassler *et al.* [5] considered side-by-side following within the application of a robotic wheelchair following a human. The authors apply a velocity-based prediction of the human's position and control both position and velocity of the follower robot. Also Kuderer and Burgard [6] considered the wheelchair application and developed an approach to predict the human trajectory taking into account information about obstacles in the environment. The authors compute the robot's trajectory so that the distance to the desired relative position along the predicted path is minimized. In this way, the robot can act foresightedly and local minima resulting from obstacles in the environment are avoided.

Related approaches predict human trajectories to generate robot motions that do not interfere with people. Bennewitz *et al.* [7] proposed to predict human trajectories based on learned motion patterns to avoid interferences in tight environments. Ziebart *et al.* [8] developed an approach using a Markov decision process (MDP) to learn from a previously observed set of trajectories. The authors learn a Q-table that can be used to predict the human's trajectories and avoid colliding with it. In our work, we use the method of Ziebart *et al.* to learn a MDP for prediction, however, we aim at making use of the prediction for learning how to efficiently follow a human.

Concerning the use of machine learning techniques in comparable problems, Goldhoorn *et al.* [9] proposed solving a hide-and-seek game using reinforcement learning with the aim to find a human and follow him/her to a goal location. The learning reward here relies on minimizing the distance between the following robot and the human during the following task. Kollar and Roy [10] developed an approach that applies reinforcement learning to reduce the effect of pose uncertainty induced by certain movements of the robot and aims at minimizing the uncertainty in the belief distribution during navigation. Furthermore, Hornung *et al.* [11] presented a technique to learn actions in order to to reach navigation goals reliably and efficiently under consideration of uncertainty.

As opposed to all the approaches discussed above, we present a learning framework that enables a service robot to efficiently reach the intended final destination of the human, thereby reducing the possible inefficiency of the human target's trajectory. Using our approach, the robot is not required to follow the human target in a fixed distance, instead, the robot predicts the navigation goal, continuously updates it, and adapts its actions.

III. Motion Prediction

In this section, we present the motion prediction technique applied in our learning framework. We predict the human's trajectory based on the part of the trajectory observed so far and use the prediction in the state space representation of our learning approach for generating efficient navigation actions for the robot.

Our technique for motion prediction is based on the work of Ziebart *et al.* [8]. This approach models the sequence of motion actions performed by a human as a softened Markov decision process (MDP) whose state space corresponds to the cells of a discretized grid map of the environment. The authors propose to train a prediction model using a softened version of the Bellman equation and value iteration to get a Q-table that represents the most likely motion action performed by the human at a certain position. This softened version uses the soft-maximum function instead of the ordinary maximum to be able to reason about the distribution of the trajectories, instead of having just one optimal trajectory. The soft-maximum function is defined as

$$softmax_x f(x) \quad = \quad \log \sum\nolimits_x e^{f(x)} \tag{1}$$

and is used within the computation of the state and action values $V(s)$ and $Q(s,a)$:

$$Q(s,a) \quad = \quad R(s,a) + V(T(s,a)) \tag{2}$$

$$V(s) \quad = \quad softmax_a \; Q(s,a) \tag{3}$$

Here, s and a represent the human's current state and corresponding action, respectively, $T(s,a)$ is the transition

function, and $R(s,a)$ is the reward after executing action a at the current state s. Eq. (2) and Eq. (3) are used to train the motion predictor based on a set of training trajectories with a reward function that takes into account obstacle locations. Ziebart *et al.* use the obtained Q-table to predict the future final destination of a partially observed trajectory as explained in the following.

Let $\zeta_{A\to B}$ denote the observed partial trajectory of the human from the initial state A to the current state B and $\zeta_{B\to C}$ the future trajectory from B to the unknown final destination state C. The probability $P(dest\ C|\zeta_{A\to B})$ of a certain destination C given a partially observed trajectory $\zeta_{A\to B}$ can then be computed with Bayes' rule, where the likelihood $P(\zeta_{A\to B}|dest\ C)$ intuitively depends on the ratio of the reward of $\zeta_{A\to B}$ and the expected value of $\zeta_{B\to C}$ to the value of the whole trajectory to the destination $\zeta_{A\to C}$:

$$\begin{aligned} & P(dest\ C|\zeta_{A\to B}) \\ & \overset{\text{Bayes'}}{=} \frac{P(\zeta_{A\to B}|dest\ C)P(dest\ C)}{P(\zeta_{A\to B})} && (4) \\ & = \frac{\frac{e^{R(\zeta_{A\to B})+V(B\to C)}}{e^{V(A\to C)}}P(dest\ C)}{\sum_D \frac{e^{R(\zeta_{A\to B})+V(B\to D)}}{e^{V(A\to D)}}P(dest\ D)} && (5) \end{aligned}$$

Here, D corresponds to a destination among the set of possible final destinations according to the training trajectories and the prior distribution $P(dest\ D)$ is known from the training set. Furthermore, the reward of a partial trajectory $R(\zeta)$ is the sum of all individual rewards of state-action pairs according to ζ:

$$R(\zeta) = \sum_{(s,a)\in\zeta} R(s,a) \quad (6)$$

and $V(X\to Y)$ denotes the softmax value function of the trajectory form state X to state Y.

Using the above equations, we can then compute the posterior over the future trajectory $\zeta_{B\to C}$ to the unknown, future destination C given the partial, observed trajectory $\zeta_{A\to B}$ using marginalization:

$$\begin{aligned} & P(\zeta_{B\to C}|\zeta_{A\to B}) \\ & = \sum_D P(\zeta_{B\to C}|\zeta_{A\to B}, dest\ D)P(dest\ D|\zeta_{A\to B}) && (7) \\ & = P(\zeta_{B\to C}|dest\ C)P(dest\ C|\zeta_{A\to B}) && (8) \\ & = \frac{e^{R}(\zeta_{B\to C})}{e^{V(B\to C)}}P(dest\ C|\zeta_{A\to B}) && (9) \\ & = e^{R(\zeta_{B\to C})-V(B\to C)}P(dest\ C|\zeta_{A\to B}) && (10) \end{aligned}$$

As can be seen, the posterior probability $P(dest\ C|\zeta_{A\to B})$ is used to weight the conditional probability $P(\zeta_{B\to C}|dest\ C)$ of the expected future trajectory given the destination.

In our implementation, we assume that the human can move one step at a time step in any of the eight possible directions corresponding to the neighbor cells or remain in the same state. Accordingly, our action space consists of nine actions. We set the reward to be inversely proportional to the distance from obstacles within a certain range of the human.

Using the learned posterior over the future trajectory according to Eq. (10), we can predict the position of the human at a certain time step given its trajectory history. How we use this information in our learning framework for the navigation actions is described in the following section.

IV. Learning Navigation Actions

A. Reinforcement Learning

We apply Sarsa(λ) reinforcement learning based on previously executed actions $a_t \in \mathcal{A}$ in different states $s_t \in \mathcal{S}$ and the obtained reward $r_t \in \mathcal{R}$. During each learning episode, the value of each state-action pair $Q^\pi(s,a)$ following the policy π is updated according to the achieved reward R_t with

$$R_t = \sum_{i=t+1}^{T} r_i \quad (11)$$

as follows [12]:

$$Q^\pi(s_t,a_t) = E_\pi\{R_t|s_t,a_t\} \quad (12)$$

Furthermore, we use an ε-greedy action selection policy.

B. State Space $\mathcal{S}$

In our model, we use a discretized representation of the environment in form of a grid map. This grid map includes static obstacles as occupied cells where the human and the robot cannot move. The state representation of the reinforcement learning framework includes the relative distance between the current 2D position of the human $\mathbf{x}_t^h$ and the current 2D position of the following robot $\mathbf{x}_t^r$ at time t as well as the relative distance between $\mathbf{x}_t^r$ and the predicted position of the human after i further time steps $\mathbf{x}_{t+i}^h$

$$s_t = \begin{bmatrix} \mathbf{x}_t^h - \mathbf{x}_t^r \\ \mathbf{x}_{t+i}^h - \mathbf{x}_t^r \end{bmatrix}. \quad (13)$$

Here, we compute $\mathbf{x}_{t+i}^h$ as explained in Sec. III. Note that we use relative positions instead of global positions to be able to generalize and learn from different trajectory classes and different maps.

C. Action Set $\mathcal{A}$

The action space consists of a set of discrete moving actions in the eight main directions in addition to standing still. Accordingly, the Q-table computed by our framework contains entries $Q(s,a)$ for each state-action pair where s is defined as in Eq. (13) and a is one of the nine navigation actions.

Note that a navigation action can only be executed if the distance to an obstacle is larger than a certain range. In this way, obstacles are implicitly modeled in our framework.

D. Reward $\mathcal{R}$

We designed the reward function so as to combine both the shortest path to the predicted human position as well as the difference between the distance traveled so far by the human and the traveled distance by the follower robot. The first term aims at staying close to the human, whereas the second term is for preferring shorter paths during the task. Accordingly, we define the intermediate reward r_t at time t as follows

$$r_t = \begin{cases} 10,000 & \text{if } t = T \\ -dist_{A^*}(\mathbf{x}_t^r, \mathbf{x}_{t+i}^h) + (trav_t^h - trav_t^r) & \text{otherwise,} \end{cases} \quad (14)$$

where T is the final time step and $dist_{A^*}$ is a function that applies the A* algorithm on the grid map representation of the environment to compute the length of the shortest path between the current robot position $\mathbf{x}_t^r$ and the predicted position of the human $\mathbf{x}_{t+i}^h$ under consideration of the obstacles. Furthermore, $trav_t$ refers to the distance traveled until time step t. This reward function leads to the generation of efficient navigation actions that minimize the cost of the robot to follow the human. The final state is reached when the robot's pose is sufficiently close to the destination the human moved to.

V. Experimental Results

A. Environment Setup

We evaluated our approach in simulation and real-world experiments in an environment of the size of 4.5 m×3.5 m containing three possible destinations reachable from the starting position of the human (see Fig. 2). We used a map resolution of 60 cm, which we found to yield reasonable navigation actions for the real robot with a diameter of 45 cm[1]. Currently, we assume perfect knowledge about the robot as well as the human position and that both are moving approximately with the same velocity[2].

B. Random Generation of Training and Test Sets

We randomly generated human trajectories for both the training and test phases composed of straight line segments of 25 cm length. The orientation of the segments relative to the destination is chosen uniformly from the interval $[0°, 60°]$. Our training set consists of 60 trajectories, 20 for each of the three possible destinations (see Fig. 2). For the test set, we randomly generated 15 trajectories for the three possible destinations as described above.

C. Parameters for the Learning Framework

During the learning of the Q-table, we use a value of 0.4 for ε of the greedy action selection to allow for exploring the state space strategy. For the execution, we decrease this value to 0.05, i.e., the robot chooses the action with the highest Q-value with probability 0.95. The motivation behind considering random actions at all is to be able to escape from potential local minima. The Q-table for our experiments was learned from 12,000 learning episodes. We consider a learning episode as successful if the distance of the robot to the human destination is smaller than 1.2m within a maximum number of 100 time steps, otherwise the episode is aborted. In the test runs, we abort the execution in case the robot does not reach the human destination within 20 time steps after the human arrives there. We used a value of 3 for i in Eq. (13) and Eq. (14), which means that our model performed a prediction three steps ahead.

In the learning episodes and test runs, we generated the robot's starting position randomly within a range of 60 cm around the human target's initial position.

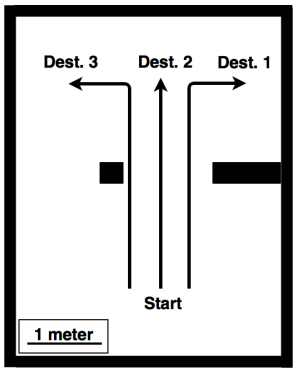


Fig. 2. Grid map of the environment with three possible destinations and corresponding human trajectories' classes used for training the motion predictor.

D. Evaluation Metrics

In order to evaluate our approach, we computed the saving w.r.t. the path length by comparing the distance traveled by the robot according to our learned navigation actions to that of a greedy following method. Here, the robot observes the humans position at each time step and starts to follow the shortest path to this observed position until the next time step. Furthermore, we evaluated the completion time of our approach to a 'wait-and-observe-first' strategy, where the robot waits until the human reaches its destination and only then starts moving according to the shortest path to the destination. We computed the shortest path on the grid map (60 cm resolution) with A* using eight possible actions leading to the neighbor cells. After that, we applied a *statistical z-test* on these metrics of the testing runs to ensure the statistical significance of our results.

E. Experiments in Simulation

For each of the human trajectories in the test set, we generated the robot's initial position randomly as explained above and performed 100 runs with different starting positions of the robot. As can be seen from Table I, our approach achieves an average gain of 7% compared to the trajectory length of the naive following strategy.

Furthermore, we evaluated our method on a test set that additionally contains a percentage of 25% non goal-directed

[1]Note that even if the environment only consists of 48 grid cells, the size of the actual state space is much bigger since we consider the relative distance of the current and predicted robot's and the human's position (see Eq. (13)).

[2]Note that adding the velocity as a further dimension into the state representation would lead to a significantly more complex learning problem.

trajectories to show the effectiveness of the robot behavior generated by our learning framework. In the corresponding runs, the human trajectory leads around the obstacle in the left part of the map (similar to the trajectory in Fig. 5). This can be seen as the case where the human fetches some items and then continues walking to its actual destination. If such scenarios are included in the test set, we even achieve an overall average gain of 18.3% (see Table I). Note that trajectories containing cycles were *not* included in the training of the Q-table for the navigation actions.

As these results show, our method yields significant distance savings, especially in the case of trajectories were the human does not move along the shortest path to his/her destination. We also achieved a shorter completion time on average compared to the 'wait-and-observe-first' strategy as can be seen from Table I.

In 5.55% of 2,000 testing runs, the execution was aborted since the robot was caught in local minima and did not reach the destination in time. The corresponding runs are not included in the evaluation.

In addition to the experiments described above, we performed a further experiment to illustrate the strength of our approach. Here, the human was walking to an unanticipated intermediate destination to pick up an item. This unexpected behavior was correctly handled by the robot as it did not follow the human but already started moving in the direction of the possible final destinations, before waiting for the human to infer the correct destination (see Fig. 3).

F. Experiments with a Real Robot

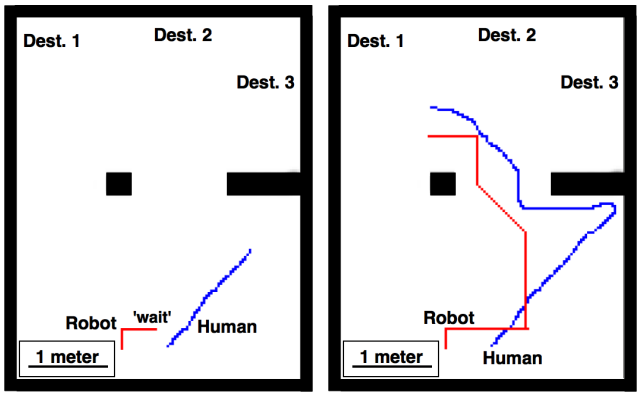
We also carried out experiments with a real robot (Robotino by Festo) to test the performance of our learning framework. In order to detect the position of the human and localize the robot, we used an external motion capture (MoCap) system.

In these real-world experiments, we focused on a scenario that shows the strength of our proposed framework, i.e., the case when the trajectory of the human does not directly lead to his/her destination. As shown in Fig. 4, the human performed an inefficient trajectory by walking around the obstacle in the left part instead of directly going to his/her destination. As can be seen, the follower robot ignored this cycle and only proceeded to follow the human afterwards. The corresponding path (see Fig. 5) is much shorter than a trajectory resulting from directly following the human.

TABLE I

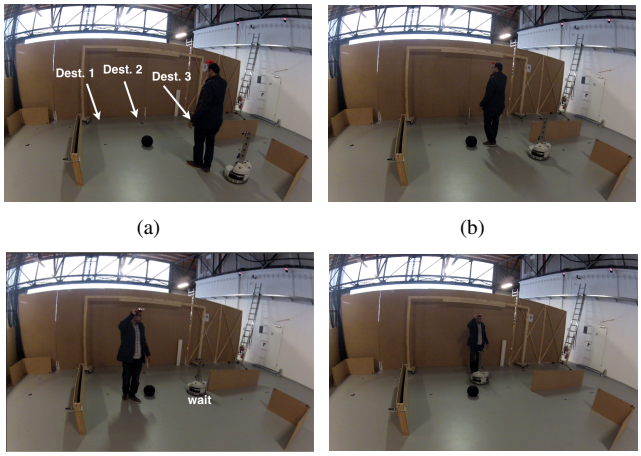
GAIN IN TRAVELED DISTANCE COMPARED TO THE NAIVE FOLLOWING STRATEGY AND GAIN IN COMPLETION TIME COMPARED TO 'WAIT-AND-OBSERVE-FIRST

Test set	Distance savings		Time savings	
	Gain	P-value	Gain	P-value
Basic test set	7%	0.0018	5.58%	0.24
Cycles included	18.3%	0.0005	8.95%	0.08



(a) (b)

Fig. 3. Experiment in which the human first walks to an unanticipated intermediate destination to pick up an item. (a) The robot does not follow the human but waits. (b) Only as the human continues moving to one of the predicted destinations, the robot follows to arrive efficiently at the final destination.



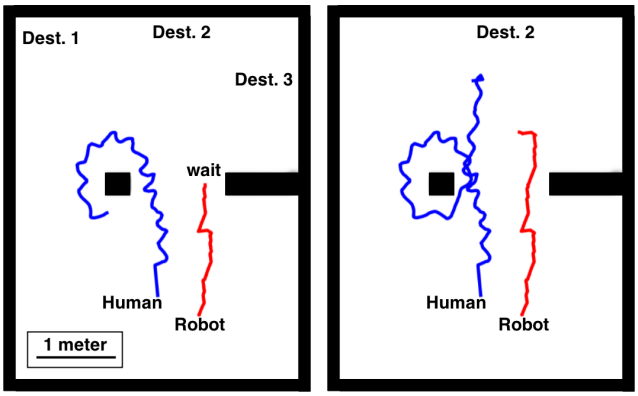
(a) (b)

(c) (d)

Fig. 4. (a) The human walks in straight line towards its (unknown) destination and the robot executes navigation actions to follow. (b) The human reaches the obstacle and walks in a cycle around it. (c) However, the robot behaving according to our learned policy ignores this cycle and waits. (d) The human continues walking towards its destination, followed by the robot.

VI. CONCLUSIONS

We developed an approach to generating efficient navigation behavior of an assistance robot. We consider the scenario in which the human moves between different destinations where it may need the help of the robot. Thus, the task of the robot is to reach these places while minimizing trajectory length and completion time. Our framework relies on a learned prediction model for the human motions that is used to reason about its future trajectory and target destination. Based on this prediction and the current robot position, we use reinforcement learning to generate the optimal navigation action for the robot. As the experiments carried out in simulation and with a real mobile robot show, the paths computed by our approach are significantly shorter and lead to a significantly reduced completion time compared to naive



(a) (b)

Fig. 5. Trajectories of the human and the robot corresponding to the experiment shown in Fig. 4 recorded by the MoCap system. (a) The robot's trajectory is rather efficient, as the robot correctly predicts that the human will continue moving to a destination in the area on top and therefore waits for the human. (b) Both human and robot resume the path to the destination.

following strategies.

In the future, we will extend our system to use the robot's on-board sensors for people tracking, which will lead to occlusions and uncertainty about the human's position. So the robot will need to learn to perform active re-localization of the human given the predictions of the human's motion.

REFERENCES

[1] 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.

[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] B.-S. Choi, J.-W. Lee, and J.-J. Lee, "Effective target-following schemes for indoor mobile robots," in *Proc. of the Int. Conf. on Information and Automation*, 2010, pp. 666–671.

[5] E. Prassler, D. Bank, B. Kluge, and M. Hägele, "Key technologies in robot assistants: Motion coordination between a human and a mobile robot," *Transactions on Control, Automation and Systems Engineering*, vol. 4, no. 1, 2002.

[6] 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.

[7] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, "Learning motion patterns of people for compliant robot motion," *Int. Journal of Robotics Research (IJRR)*, vol. 24, no. 1, 2005.

[8] 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, pp. 3931–3936.

[9] 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.

[10] T. Kollar and N. Roy, "Using reinforcement learning to improve exploration trajectories for error minimization," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2006.

[11] A. Hornung, M. Bennewitz, and H. Strasdat, "Efficient vision-based navigation – Learning about the influence of motion blur," *Autonomous Robots*, vol. 29, no. 2, 2010.

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