Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Motion Planning for People Tracking in Uncertain and Dynamic Environments

Tirthankar Bandyopadhyay, Nan Rong, Marcelo Ang, David Hsu, Wee Sun Lee

***Abstract*—Target tracking is an important capability for autonomous robots. The goal of this work is to construct *motion strategies* for a robot so that it can handle visual and mobility obstruction due to obstacles and maneuver effectively to track a mobile target in a dynamic, uncertain environment. There are two broad approaches to address dynamic changes and uncertainties in the environment: to react fast or to plan ahead. The choice often depends on the amount of prior information available on the environment and the target behavior. This paper gives an overview of our work on target tracking using these two approaches.**

First, we present a greedy algorithm. It uses purely local geometric information from the robot's sensor to compute the robot's motion at each time step, and yet carefully balances the robot's ability to track the target in both the current and the future time. The algorithm uses only information from the robot's sensor and requires no prior information on the environment or the target behavior. This has been shown to work well on a real robot with a 2-D laser sensor in a crowded school cafetaria.

Second, we use partially observable Markov decision process (POMDP) to build a model of target behavior. As a result, the robot is capable of more sophisticated tracking behavior. For example, it may intentionally allow the target to get out of sight in order to minimize its own movement and save energy, but does not compromise long-term tracking performance. This is ongoing work and we show simulation results demonstrating the effectiveness of the approach.

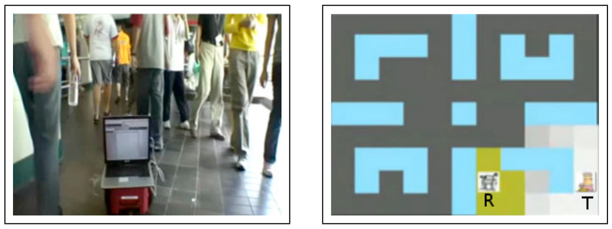
I. Introduction

Target tracking has many applications. In home care settings, a tracking robot can follow elderly people around and alert caregivers of emergencies. In security and surveillance systems, tracking strategies enable mobile sensors to monitor moving targets in cluttered environments. In this paper, we focus on developing motion strategies for a robot equipped with visual sensors so that it can effectively track and follow a moving target, despite obstruction by obstacles. Target identification, an important component of target tracking is assumed.

Just as in classic motion planning [13], we must consider *motion constraints* resulting from both obstacles in the environment and the robot's mechanical limitations. In particular, the robot must not collide with obstacles. Target following has the additional *visibility constraints* due to sensor limitations, *e.g.*, obstacles blocking the view of the robot's camera. Both motion constraints and visibility constraints play a significant role for target following in cluttered and dynamic environments.

The robot can address dynamic changes and uncertainties in the environment either by reacting fast to each changes or by modeling these uncertainties and planning ahead. The choice depends on the availability of prior information to the robot. When the environment or the target behavior is unknown, the robot has to plan its motion based on just the local information available and try to maximize the duration for which it can keep the target in view. On the other hand, when the environment is known and the target behavior can be modeled, the robot can incorporate this information to generate sophisticated motion strategies that maximizes the overall time that the target is in view.

This paper gives an overview of our work in following the target using these two approaches. In the rest of this section, we motivate and describe the approaches in light of two concrete examples (Figure 1).



(*a*) (*b*)

Fig. 1. Different scenarios: (a) Crowded canteen environment : Highly dynamic and unknown environment, suitable for local planning (b) Home care application : Uncertainty in target's position handled by POMDP tracker can generate sophisticated behaviors.

Let us take a specific scenario of an automated personal shopping assistant following an elderly person in a shopping mall, or keeping an eye on young kids while their parents shop. The shopping mall is a complex environment. People walking around add to the visual occlusions and motion obstructions, thereby creating a highly cluttered and dynamic environment (Figure 1*a*). While the layout of the environment might be available in some cases, exact maps for localizing the robot are hardly provided. On top of that, the target can be completely unpredictable in moving from one shop to another. In such situations where little is known about the target behavior or the environment, a local greedy strategy is more effective than complete planning.

Key to our algorithm is the definition of a risk function, which tries to capture the targets ability in escaping from the

Tirthankar Bandyopadhyay is with CENSAM IRG, SMART tirtha@smart.mit.edu
Nan Rong is a PhD. student at CMU nan.rong@gmail.com
Marcelo Ang is Associate Professor in Department of Mechanical Engineering, National University of Singapore, mpeangh@nus.edu.sg
David Hsu and Wee Sun Lee are Associate Professors in Department of Computer Science, National University of Singapore, dyhsu@comp.nus.edu.sg,leews@comp.nus.edu.sg

robot sensors visibility region in both short and long terms. To select actions effectively, the robot must balance between the short-term goal of preventing the immediate loss of the target and the long-term goal of keeping it visible for the maximum duration possible. Interestingly, a good compromise can be achieved, using only local information available to the robots sensors. By analyzing the local geometry, our algorithm computes a global risk function as a weighted sum of components, each associated with a single visibility constraint. It then chooses an action to minimize the risk locally in a greedy fashion.

As the algorithm uses only local geometric information available to the robots visual sensors, it does not require a global map and thus bypasses the difficulty of localization with respect to a global map. Furthermore, uncertainty in sensing and motion control does not accumulate. This improves the reliability of tracking. We have tested the algorithm in a crowded school cafeteria at lunch time. The crowd of students moving towards food stalls and then towards their seats create a truly dynamic and cluttered environment. Our implementation shows that the tracker is robust to temporary occlusions and in uneven terrain. The algorithm scales well with high clutter and obstructions and shows good performance for reasonable target behavior.

On the other hand, if enough information is available on the environment and the target behavior, the prior information can be used by the tracker to come up with 'smarter' strategies to improve the tracking performance. We formulate the tracking problem into a POMDP framework. POMDP trackers integrate global information on the target behavior and the environment for optimal decision making. Let us take a specific scenario from the homecare application. Imagine that an elderly person moves around at home and has a call button to call a robot over for help (Figure 1*b*). The call status stays on for some time and then goes off. If the robot arrives while the call status is on, it gets a reward; otherwise, it gets no reward. Clearly, the robot should stay close to the person in order to improve the chance of receiving rewards, but at the same time, the robot needs to minimize movement in order to reduce power consumption. Moreover, there might be regions where the robot might not be allowed to follow, e.g. bathroom etc. So the naive strategy of following right behind the person does not work well. The map of the environment is available to the robot but there are uncertainties in the location of the target and the robot itself w.r.t this map.

The problem of target tracking comprises of *target searching* and *target following*. By modeling target tracking as a partially observable Markov decision process (POMDP) [20], searching and following can be unified. The main idea is to represent the target position as a probability distribution, whether the target is visible to the robot sensors or not. So the target position is always "known" to the robot with some degree of uncertainty. The robot then chooses its actions according to a probabilistic model of target behaviors and a reward function that encourages the robot to keep the target visible.

The POMDP framework offers several other advantages. It provides a principled way to deal with uncertainties in robot control and sensing. It also easily incorporates additional requirements, *e.g.*, minimizing the robot's power consumption.

We formulate the tracking problem as a POMDP and use a sampling based algorithm SARSOP [10] to generate interesting tracking behaviors, *e.g.*, anticipatory moves that exploit target dynamics, information-gathering moves that reduce target position uncertainty, and energy-conserving actions that allow the target to get out of sight, but do not compromise tracking performance.

II. Prior Work

Target tracking has received tremendous amount to attention. One important part of target tracking is to detect and identify the target(s) from noisy, error prone and uncertain sensor data. Our mention of a few passing references below, is by no means representative of the work by the community. For single targets, kalman filter [4] and particle filters [11] have been used. For multiple targets, Joint Probability Data Association Filters (JPDAF) was proposed [8] which was implemented for people tracking among others by [19]. Multi-Hypothesis Tracking (MHT) was proposed by Reid [18]. An interesting and quite recent work on leg tracking has been described in [1].

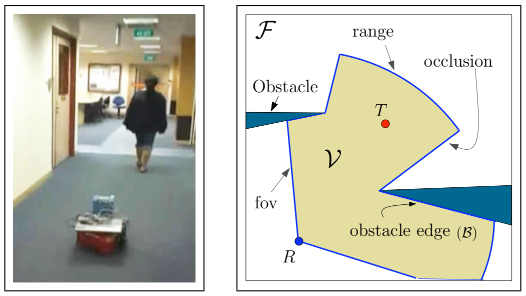
Motion strategies for target tracking depend on the amount of information available. If both the environment and the target trajectory are completely known, optimal target following strategies can be computed through dynamic programming [14], or by piecing together certain canonical curves [6], If only the environment is known, one can preprocess the environment by decomposing it into cells separated by critical curves. The decomposition helps to identify the best robot action as well as to decide the feasibility of tracking [15]. If the environment and the target trajectory are both unknown in advance, one approach is to move the robot so as to minimize an objective function that tries to capture the short- and long-term risk of losing the target [3], [9], [16]. With few exceptions [7], Most of these approaches do not handle uncertainties in robot control and sensing. Other probabilistic approaches to target tracking include, *e.g.*, [21].

Our POMDP tracking problem is related to the Tag problem described in [17]. However, the problems considered here involve a much larger number of states and more complex target behaviors. The SARSOP algorithm is also more efficient than the PBVI algorithm used in [17] and can handle more realistic target tracking tasks.

Another potential difficulty with the POMDP approach is the acquisition of a good probabilistic model of target behavior, but machine learning techniques can help [5].

III. Local Greedy Tracker

For an unknown environment and an unknown target behavior, the robot must execute an online reactive strategy that takes into account only local information. In this work, to identify and track a person, we use visibility based sensors, based on



(a) (b)

Fig. 2. Formulation of the target tracking problem into geometrical parameters extracted from local information

the standard straight line-of-sight visibility model (Figure 2b). In the free space $\mathcal{F}$, the *visibility set* $\mathcal{V}(x)$ is given by,

$$\begin{aligned}\mathcal{V}(x) \quad = \quad & \{q \in \mathcal{F} \mid \overline{xq} \subset \mathcal{F} \text{ and } d(x,q) \leq D_{max} \text{ and} \\ & \theta_{min} \leq ang(x,q) \leq \theta_{max}\}\end{aligned}$$

where $d(x,q)$ denotes the distance between x and q, while $ang(x,q)$ is the orientation of q w.r.t. x. Information about the local environment is encoded into the boundary ($\partial\mathcal{V}$), of the visibility polygon ($\mathcal{V}$).

Both the robot and the target's motion, are formulated with a simple discrete-time constant velocity model. As the target behavior is unknown, its velocity ($\mathbf{v}'$) is modeled by a gaussian around its current heading : $\mathbf{v}'(t+\Delta t) = \mathcal{N}(\mathbf{v}'(t), \sigma)$. The variance σ gives a measure of confidence in estimating the target velocity. Although we use a Gaussian distribution to model the uncertainty in the target behavior, the approach remains valid for any other velocity prediction method, even non-parametric ones.

A. Local Greedy Approach Overview

The objective of the robot is to keep the target inside the robot's visibility, $\mathcal{V}$, for as long as possible. The target can escape $\mathcal{V}$ through its boundaries that lie in free space. We term these boundaries as *escape edges* (Figure 2b). Since the robot has no control over the enviornment or the target's motion, it can only prevent the target's escape by manipulating the *escape edges*, $\{\mathcal{G}_i\}$ away from the target. The ability of the robot to manipulate $\mathcal{G}_i$ effectively is important in maintaining the target in view. Let us denote the manipulation ability of the robot for a single escape edge, $\mathcal{G}_i$, by the symbol, $\Delta\mathcal{G}_i$. $\Delta\mathcal{G}_i$ is a function of the robot position, $\mathbf{x}$, and its actions, $\mathbf{v}$: $\Delta\mathcal{G}_i(\mathbf{x}, \mathbf{v})$. The risk of losing the target, on the other hand, depends on : (a) the target position ($\mathbf{x}'$), (b) the relative target velocity ($\mathbf{v}'$) w.r.t. to $\{\mathcal{G}_i\}$, and finally (c) the robot's ability to manipulate the edges, $\Delta\mathcal{G}_i$. We can then formulate a risk function (Φ) and choose the robot action, $\mathbf{v}^\star$, to minimize Φ:

$$\begin{aligned}Risk \quad &= \quad \Phi(\mathbf{x}', \mathbf{v}', \{\mathcal{G}_i\}, \{\Delta\mathcal{G}_i(\mathbf{x},\mathbf{v})\}) \\ \mathbf{v}^\star \quad &= \quad \textbf{arg min} \;\; \Phi(\mathbf{x}', \mathbf{v}', \{\mathcal{G}_i\}, \{\Delta\mathcal{G}_i(\mathbf{x},\mathbf{v})\}) \qquad (1)\end{aligned}$$

While Φ is the risk of losing the target through any escape edge in the entire $\mathcal{V}$, we can assign a risk φ_i, of losing the target to each escape edge, $\mathcal{G}_i$. We approximate the total risk Φ, by the expected risk for all the gaps.

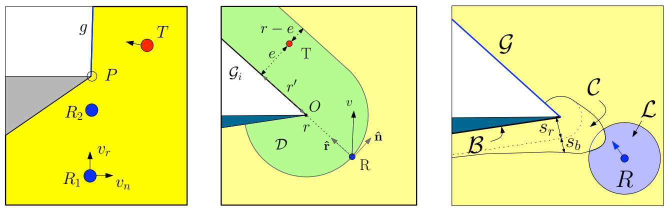
$$\Phi \approx E[\varphi_i] = \sum_i p_i \varphi_i(\mathbf{x}', \mathbf{v}', \mathcal{G}_i, \Delta\mathcal{G}_i(\mathbf{x},\mathbf{v})), \mathbf{v}^\star \approx \sum_i p_i v_i^\star \qquad (2)$$

where p_i is the probability of the target's escape through $\mathcal{G}_i$. p_i is computed based on the target's current velocity, $\mathbf{v}'$. The details can be found in [3].

However, in choosing $\mathbf{v}^\star$, the robot has to satisfy many constraints on the desired robot positions, *e.g. obstacle avoidance* considerations or on the robot's actions like *kinematic, dynamic constraints*. We define a *feasible region*, $\mathcal{L}$ (x), that satisfies all the constraints ($\mathcal{C}_i(x)$) in the position domain : $\mathcal{L}(x) = \bigcap_i \mathcal{C}_i(x)$. The local greedy optimization then chooses an action ($\mathbf{v}^\star$), that minimizes Φ while satisfying $\mathcal{L}$ (x) in the time step Δt,

$$\mathbf{v}^\star = \textbf{arg min} \;\; \Phi(\mathbf{x}', \mathbf{v}', \{\mathcal{G}_i\}, \{\Delta\mathcal{G}_i(\mathbf{x},\mathbf{v})\}), s.t \quad \mathbf{v}^\star\Delta t \in \mathcal{L}(\mathbf{x}) \qquad (3)$$

B. Risk Formulation : φ



(a) (b) (c)

Fig. 3. (a) Relative position determines risk, (b) Local geometric parameters, (c) Obstacle Dilation

For successful tracking, the robot must balance the short-term goal of preventing the immediate loss of the target through these escape edges and the long-term goal of maximizing the duration of tracking in the future. Let us look at a simple 2-D example shown in Figure 3a.

For the robot positioned at R_1, the obstacle (the dark-colored triangle) creates an occlusion edge g with one endpoint at P. The robot has the short-term goal of preventing the target T's escape through g at the current instant. It achieves this by swinging g away from the target, using velocity v_n. The robot's longer-term goal is to move towards P using velocity v_r, because it can eliminate the occlusion edge g completely when it reaches P. Since the robot's maximum speed is bounded by V, there is a trade off in choosing the velocity components v_r and v_n. Clearly, this trade off depends on the relative positions and velocities of the target and the robot w.r.t P. For example, the robot at position R_2 can afford a higher v_r, as the shortest distance from the target to g is greater than that of the robot and there is no immediate risk of losing the target. Whereas at R_1, the target is closer to g than the robot, and the short-term goal of preventing the loss of target becomes much more important.

We formulate a risk function that incorporates this trade off between the current and future risk in terms of local geometrical parameters (Figure 3*b*).

In the previous work [3], a local greedy algorithm based on *relative vantage* was proposed. Relative vantage refers to the ability of the robot to eliminate $\mathcal{G}_i$ before the target can escape through it. We introduce a region around $\mathcal{G}_i$, called *vantage zone* as, $\mathcal{D} = \{q : q \in \mathcal{V}; dist(q, \mathcal{G}_i) \leq dist(\mathbf{x}, \mathcal{G}_i)\}$

The objective of the robot is to keep the targets away from $\mathcal{D}$ and accordingly, we can take the measure of time taken ($t_{r.v}$) to move the target outside $\mathcal{D}$, as the risk value. From Figure 3*b*,

$$\varphi_g = t_{r.v} \approx \frac{dist(\mathrm{t}, \mathcal{D})}{rel.vel(\mathrm{t}, \mathcal{D})} \approx \frac{r - e}{v_{\text{eff}}}, \qquad v_i^\star = \frac{\varphi_g}{v_{\text{eff}}}\left(\frac{r'}{r}\hat{\mathbf{n}} + \hat{\mathbf{r}}\right)$$

where $v_{\text{eff}} = v_r + v_n(r'/r) - v'_{\text{e}}$ is the effective velocity in the direction along the shortest path from the target to $\mathcal{G}_i$.

Similar considerations can be applied to the field of view (FoV) limits and the range limits. Derivation of these special cases are omitted due to space limitation. The reader is pointed to [2] for details.

C. Obstacle Avoidance

Although, purely low-level reactive obstacle avoidance techniques, can handle dynamic and unknown environment, they may sometimes move the robot contrary to the required tracking direction. On the other hand, planning in the configuration space may be too computationally expensive in a cluttered and dynamic environment. We propose a local obstacle avoidance method with a small look-ahead. The robot's velocity is used to enlarge the obstacle edges. These extended obstacles then constrain the planned robot motion.

First, we approximate the robot's size by the radius (s_r) of its bounding circle. Then, we compute the finite braking distance, s_b, using the maximum decceleration and the robot's current velocity. This braking distance, s_b and the robot's dimension, s_r, defines a collision region $\mathcal{C}$ (x), around the obstacle edge, $\mathcal{B}$, Figure 3*c*,

$$\mathcal{C}(x) = \{q \in \mathcal{V} : d(q, \mathcal{B}) \leq (s_r + s_b)\} \tag{4}$$

The robot can actively change the shape of $\mathcal{C}$ by changing its speed and heading. For safe navigation, the robot must avoid $\mathcal{C}$. If we denote the *reachable* region of the robot in Δt, as $\mathcal{R}$, the feasible region becomes $\mathcal{L} = \mathcal{R} - \mathcal{C}$. As an example, assuming omni-directional motion ability of the robot with no dynamics in Figure 3*c*, $\mathcal{R}$ is a disk of radius, $V\Delta t$, and the darker shaded region shown is $\mathcal{L}$. Appropriate motion models, non-holonomic constraints, motion dynamics *etc* change the shape of $\mathcal{R}$, but the basic approach remains the same.

We substitute the details of the escape edge risk and the obstacle avoidance constraints in expression

$$\mathbf{v}^\star = \sum_i p_i v_i^\star \quad s.t \quad \mathbf{v}^\star \Delta t \in \mathcal{L}(\mathbf{x}) \tag{5}$$

D. Experimental Results

The tracking algorithm is implemented on a Pioneer P3-DX differential drive robot. A SICK-lms200 range sensor is mounted on the robot. The laser returns 361 readings on a field of view of 180deg at the resolution of 0.5deg. The maximum range of the sensor is 8m. The control algorithm runs on a Pentium M Processor @1.5GHz laptop running Player server v-2.0.5 on linux. The algorithm runs at 10Hz. Implementation details are described in [2].

In the following we show a comparison of our algorithm with visual servo algorithm. Subsequently we showcase two of our experimental runs that was performed in the school cafeteria. The videos of these and more experiments performed on indoor, canteen and outdoor tracking are available online at http://guppy.mpe.nus.edu.sg/~tirtha/research/Hardware/hardImpl.html. More detailed analysis and comparison to existing algorithms is available in [2].

1) Comparison with Visual Servo (Figure 4 & Figure 5) : In Figure 5, a box is pushed between the target and the robot to occlude the target. The responses of a simple visual servo algorithm is compared to the vantage tracker. Since, the vantage tracker actively tries to avoid possible future occlusions, it is able to adapt to the changing environment (Figure 5*b-1*). A point to note is that the vantage tracker does not model the motion of the environment but just replans its motion at a high frequency, making the tracker independent of the dynamic nature of the environment. Later, when the box stops and the target starts to move (Figure 5*c*), the tracker is able to successfully follow the target (Figure 5*d*). In comparison, the simple visual servo tracker does not model the dynamic environment and loses the target from its view (Figure 4).

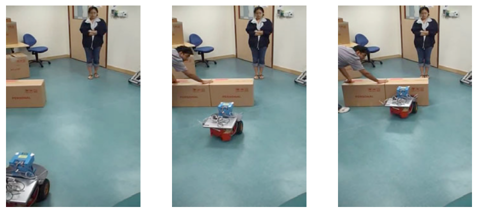

(*a*) (*b*) (*c*)

Fig. 4. Visual Servo : Since the robot does not take into account the environment information (the moving box), it moves straight ahead towards the target (b) and loses the target behind the occluding box (c).

2) Tracking in a Crowd (Figure 6) : This experiment was done during lunch hour to capture the dynamic environment of the canteen at peak rush time. The robot follows the target in grey t-shirt (1). As the target moves into the canteen area the crowd keeps increasing (2,3,4). Moreover in (3,4,5) the robot has to maneuver through a narrow pathway while avoiding incoming people and keeping the original target in view which makes following the target more difficult.

3) Visual Occlusions (Figure 7): A challenging aspect of following the target in a crowd is when someone walks in

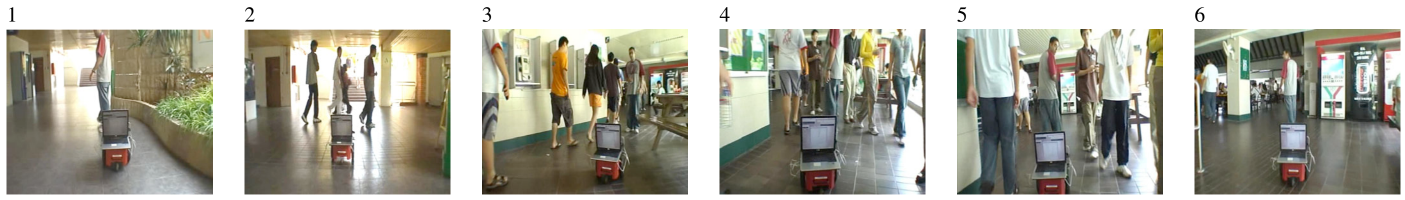


Fig. 6. Target following lunch hour rush crowd at school cafeteria

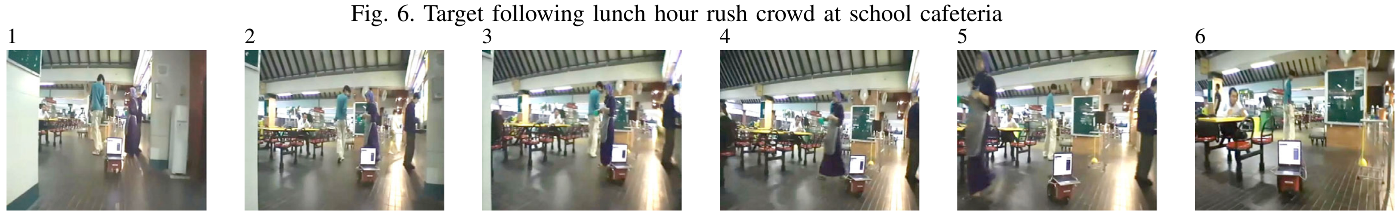


Fig. 7. Fast online local greedy algorithm is robust to temporary occlusions

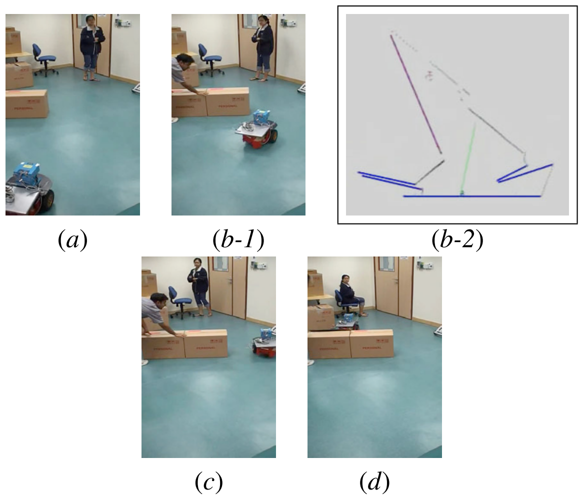


Fig. 5. Vantage tracker : (*b-2*) shows the robot's local perception of the environment. The target is marked by T, the blue lines are the occlusion edges, red line is the most critical occlusion and the green segment starting from R denotes the robot's motion decision. The robot sees the target too close to the occlusion and swings out.

between the robot and the target. In this set of snapshots, the robot is following the target in green t-shirt when it faces an temporary occlusion by a lady (in purple) walking across unexpectedly (2,3,4). The robot slows down to avoid collision (3,4) and returns to following the target when the occlusion has passed. Due to the fast online nature of the tracking algorithm, temporary occlusions in such a dynamic environment is handled well by the robot.

IV. POMDP Tracker

We start with a brief review of POMDPs. See [12] for a more complete introduction. We then describe how to model the target tracking problem as a POMDP.

A. *Background on POMDPs*

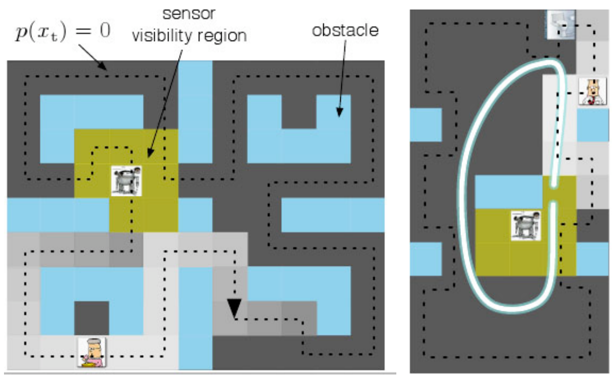
A POMDP models an agent taking a sequence of actions under uncertainty to maximize its total reward. Formally it is specified as a tuple $(S, A, \mathcal{B}, T, Z, R, \gamma)$, where S is a set of states, A is a set of actions, and $\mathcal{B}$ is a set of observations.

The agent always lies in some state $s \in S$. In each time step, it takes some action $a \in A$ and moves from a start state s to an end state s'. Due to the uncertainty in action, the end state s' is described as a conditional probability function $T(s, a, s') = p(s'|s, a)$, which gives the probability that the agent lies in s', after taking action a in state s. The agent then makes an observation on its current state. Due to the uncertainty in observation, the observation result $o \in \mathcal{B}$ is again described as a conditional probability function $Z(s, a, o) = p(o|s, a)$ for $s \in S$ and $a \in A$.

In each step, the agent receives a real-valued reward $R(s, a)$, if it lies in state s and takes action a. The goal of the agent is to maximize its expected total reward by choosing a suitable sequence of actions. In this work, we consider infinite-horizon POMDPs, in which the sequence of actions to be chosen has infinite length. We specify a discount factor $\gamma \in (0, 1)$ so that the total reward is finite and the problem is well defined. In this case, the expected total reward is $\mathrm{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)]$, where s_t and a_t denote the agent's state and action at time t.

The solution to a POMDP is an optimal *policy* that maximizes the expected total reward. Normally, a policy is a mapping from the agent's state to a prescribed action. However, in a POMDP, the agent's state is partially observable and not known exactly. So we rely on the concept of *belief state*, or *belief*, for short. A belief is a probability distribution over S. A POMDP policy $\pi\colon \mathcal{B} \to A$ maps a belief $b \in \mathcal{B}$ to the prescribed action $a \in A$.

A policy π induces a value function $V^{\pi}(b)$ that specifies the expected total reward of executing policy π starting from b. It is known that V^*, the value function associated the optimal policy π^*, can be approximated arbitrarily closely by a convex and piecewise-linear function $V(b) = \max_{\alpha \in \Gamma}(\alpha \cdot b)$, where b is a discrete vector representation of a belief and Γ is a finite set of vectors called α-vectors. Each α-vector is associated with an action, and the policy can be executed by selecting the action corresponding to the best α-vector at the current



(a) (b)

Fig. 8. Simulation experiments for target tracking.

belief b. So the policy can be represented by a set Γ of α-vectors. Policy computation, which, in this case, involves the construction of Γ, is usually performed offline.

Given an policy π, the control of the agent's actions is performed online in real time. It consists of two steps executed repeatedly. The first step is policy execution. If the agent's current belief is b, it then takes the action $a = \pi(b)$, according the given policy π. The second step is belief estimation. After the agent takes an action a and receives an observation o, its new belief state b' is given by

$$b'(s') = \tau(b, a, o) = \eta Z(s', a, o) \sum_{s \in S} T(s, a, s') b(s),$$

where η is a normalizing constant. The process then repeats.

B. Target Tracking as a POMDP

Our problem setting is motivated by homecare applications. Imagine that an elderly person moves around at home and has a call button to call a robot over for help. The call status stays on for some time and then goes off. If the robot arrives while the call status is on, it gets a reward; otherwise, it gets no reward. Clearly the robot should stay close the person in order to improve the chance of receiving rewards, but at the same time, the robot needs to minimize movement in order to reduce power consumption. So the naive strategy of following right behind the person does not work well.

When the environment information is known and the target behavior is known, we propose a POMDP tracker. To formulate the problem as a POMDP, we model the environment as a regular grid. See Figure 8 for examples. The robot and the target (in this case, the person with the call button) can occupy any of the grid cells that are free of obstacles. The state s of this POMDP is composed of the robot position x_r, the target position x_t, and the call status c: $s = (x_r, x_t, c)$. If the environment contains n free cells, then there are $n \cdot n \cdot 2 = 2n^2$ distinct states, resulting in a belief space of $2n^2$ dimensions.

In one time step, the target can stay where it is or move to a neighboring cell. The target motion is described by a given probability function T_t, conditioned on the target's current position: if the target is currently at x_t, it will be at x'_t in the next time step with probability $T_t(x_t, x'_t) = p(x'_t | x_t)$.

The person may turn on the call button in each step with probability p_1. If the call status is on, the person may turn it off with some probability p_2 in each time step, indicating that help is no longer needed. This model has two main implications. First, as the call duration follows the geometric distribution, the mean duration of a call is $1/p_2$, Second, most calls are short. The robot must arrive quickly in order to receive rewards, thus increasing the difficulty of tracking.

The robot motion resulting from an action is described similarly by another probability function T_r, conditioned on both the robot's current position and its action. The robot's actions consist of commands to stay where it is or to move to a neighboring cell. If the robot is currently at x_r and takes action a it will be at x'_r in the next time step with probability $T_r(x_r, a, x'_r) = p(x'_r | x_r, a)$. Note that the robot may not be able to execute the commands perfectly due to control uncertainty. This can be modeled with a suitable T_r.

We assume that the robot can see the target through its sensors if they lie in the same or neighboring cells. Uncertainty on the target position due to sensor noise can be modeled in the observation probability function Z.

The robot receives a reward, if it reaches the cell that the target occupies while the call button is on. In one step, if the robot does not move, it incurs no costs (*i.e.*, negative rewards). Otherwise, it incurs a cost proportional to the distance traveled. The robot's goal is to maximize the expected total discounted reward.

The POMDP formulation does not explicitly differentiate whether the target is visible or not. To execute a policy, the robot maintains a belief of the target position. When the target is visible to the sensors and the sensor data are good, the belief is sharpened. When the target is not visible or the sensor data are poor, the belief becomes more diffuse. In the extreme case, when the target remains invisible for a long time, the belief may eventually converge to a uniform distribution. This way, target searching and target following are unified in a natural way. Clearly, if the robot knows the target position well, it can choose better actions and receive higher rewards. Therefore, an optimal policy favors sharp beliefs, while also taking into account the cost of obtaining them.

C. Simulation Results

We used SARSOP [10] to compute tracking policies in several simulated environments. See Figure 8 for examples. The light blue areas in the figures indicate obstacles. The black dashed curve indicates the target's path. The target motion is non-deterministic: it follows this path, but in each time step, it may pause or proceed along the path with equal probabilities. The green area around the robot indicates the robot sensor' visibility region. The various shades of gray show the robot's belief of the target position. Lighter color indicates higher probability. To focus on target tracking behaviors, we assume in these experiments that there is no uncertainty in robot control and sensing. The robot can execute motion commands and observe its own position and call status perfectly. It can also observe the target position perfectly, if the target is visible.

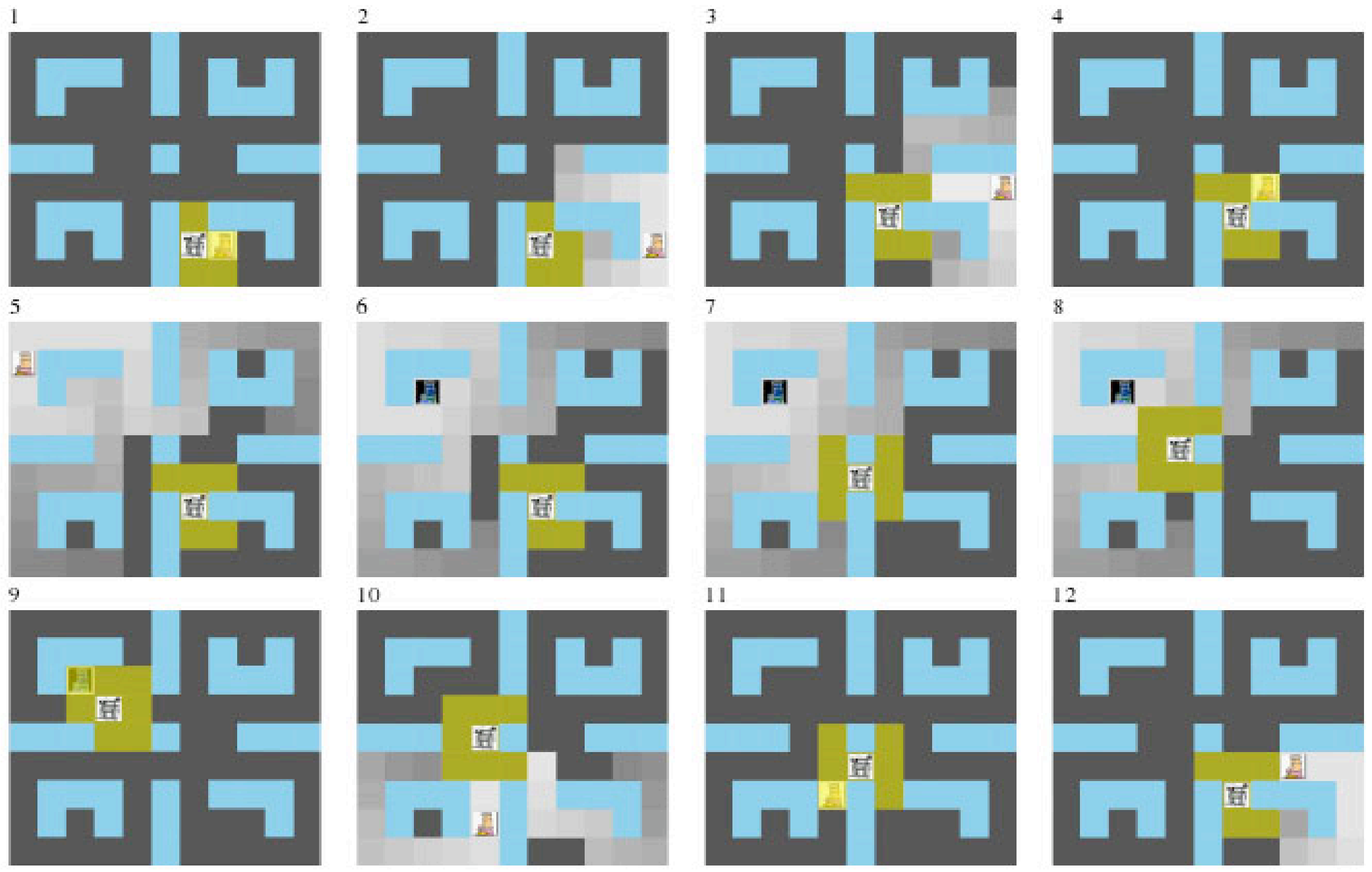


Fig. 9. Snapshots of a simulation run.

Uncertainties in control and sensing can be easily incorporated into the POMDP if needed. If the robot reaches the current target position while the call status is on, it receives a reward of 100. The robot receives a reward of -1 for a horizontal or vertical move, a reward of $-\sqrt{2}$ for a diagonal move, and a reward of 0 if it stays stationary. The discount factor is set to 0.95.

In the first experiment, we have a home-like environment (Figure 8a). The corresponding POMDP has $9,248$ states. SARSOP computed a policy in about 48 minutes. We performed several simulation runs to examine its performance and observed interesting robot tracking behaviors:

- anticipatory moves that exploit target dynamics,
- information-gathering moves that reduce target position uncertainty,
- approaching the target along a nearly optimal path when the robot is called,
- minimizing movement by allowing the target to get out of sight, but not compromising long-term tracking performance.

It is important to bear in mind that these behaviors are not manually specified, but automatically captured by the POMDP through policy computation.

Snapshots of a single simulation run are shown in Figure 9. Initially, the target lies within the robot sensor's visibility region, and the robot's belief on target position consists of a single peak (snapshot 1). As the target moves, the robot does not follow along and intentionally let the target get out of sight, in order to minimize movement and reduce energy consumption. Now, although the target is *not* visible, the robot still has the target reasonably well localized by maintaining a belief on the target position: the target is well within the high-probability region of the current belief (snapshot 2). Instead of following the target, the robot tries to anticipate the future position of the target by exploiting the target dynamics and makes a move towards this position (snapshot 3). As there is no call, the robot's move purely serves the purpose of gathering information on the target position. When the target passes by, the belief on target position is sharpened (snapshot 4). If the target is not observed for a while, the uncertainty may become large, but the robot is still able to maintain a belief that reflects the current target position well: the target is located within a high-probability region (snapshot 5). When there is a call (snapshot 6), it uses the current belief to find the region that contains the target with high probability. It then moves towards the region along the shortest path (snapshots 6–9). In general, the robot may need to search this region, but here it luckily finds the target right away and receives a high reward (snapshot 9). The robot then makes another anticipatory move to reduce target position uncertainty (snapshots 10–12). Interestingly, the robot position in snapshot 12 is exactly the same as that in snapshot 3, despite that the target positions and beliefs are quite different. It is, of course, not coincidence. This particular position guards both of the two ways into the lower right corner of the environment. By occupying this position, the robot can intercept the target as it exits the entrances without following it. The tracking behavior here reveals that the computed policy captures well the interaction between the environment geometry and the target dynamics. In this simulation run, there are 3 calls in total, and all are answered in time. The target travels a total distance of 141, while the robot about 20.

In the second experiment, the environment contains a special cell corresponding to a bathroom lying on the target's path (Figure 8*b*). After entering the bathroom, the target stays there with probability 0.95 and leaves with probability 0.05 in each step. The corresponding POMDP has $7,200$ states. SARSOP computed a policy in about 16 minutes. Roughly, to execute this tracking policy, the robot moves on the inner loop (the thick white curve in Figure 8*b*) and follows the target that moves along the outer loop (the dashed black curve in Figure 8*b*). It approaches the target directly when called.

Videos of both experiments above as well as additional experiments are available at http://motion.comp.nus.edu.sg/projects/tracking/tracking.html. We are currently performing more experiments to evaluate tracking performance quantitatively.

V. Conclusion

In this paper, we gave a brief overview of two approaches that are adept at tackling the problem of target tracking in different scenarios depending on the information available for planning. When the environment and the target behavior is unknown, an online greedy algorithm that acts based only on local information is proposed. This has been shown to work on hardware in the school cafetaria on a crowded lunch hour. We also compare this work with visual servo in a controlled setup to show the inherent improvement of the approach. On the other hand, for known environment and target models, the POMDP tracker provides more sophisticated behaviors, where it could lose the target temporarily to minimize its energy consumption while not compromising the tracking effectiveness. Simplified assumption about the sensing and motion models have been made. Early simulation results show sophisticated robot behaviors.

In this work the target identification is assumed. The target identification can be seen as a complimentary problem to the motion planning aspect of target tracking. Improved techniques for target disambiguation and development of target's motion models help in the task of target following. On the other hand, motion planning for maintaining a good view of the target aids the sensors to continously sense the target improving the identification and modeling of the target reliably. Basic constraints of the sensors like the field of view and range limitations can be incorporated into the motion strategy such that the robot always keeps the sensor facing the target. Uncertainty in the target track can be included as an objective function for the robot to minimize by planning a suitable motion strategy.

References

[1] K. O. Arras, S. Grzonka, M. Luber, and W. Burgard. Efcient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In *Proceedings of the Int. Conf. on Robotics and Automation.*, 2008.

[2] T. Bandyopadhyay, D. Hsu, and M. Ang Jr. Motion strategies for people tracking in cluttered dynamic environments. In *Proc. Int. Symp. on Experimental Robotics*, 2008.

[3] T. Bandyopadhyay, Y. Li, M. Ang Jr., and D. Hsu. A greedy strategy for tracking a locally predictable target among obstacles. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 2342–2347, 2006.

[4] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press Inc., Orlando, Florida, 1988.

[5] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun. Learning motion patterns of people for compliant robot motion. *Int. J. Robotics Research*, 24(1):3148, December 2005.

[6] A. Efrat, H. González-Baños, S. Kobourov, and L. Palaniappan. Optimal strategies to track and capture a predictable target. In *Proc. IEEE. Int. Conf. on Robotics & Automation*, pages 3789–3796, 2003.

[7] P. Fabiani, H. González-Baños, J. Latombe, and D. Lin. Tracking a partially predictable target with uncertainties and visibility constraints. *J. Robotics & Autonomous Systems*, 38(1):31–48, 2002.

[8] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE Journal of Oceanic Engineering*, OE-8(3):173184, July 1983.

[9] H. González-Baños, C.-Y. Lee, and J.-C. Latombe. Real-time combinatorial tracking of a target moving unpredictably among obstacles. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 1683–1690, 2002.

[10] D. Hsu, W. Lee, and N. Rong. A point-based POMDP planner for target tracking. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 2644–2650, 2008.

[11] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *Int. J. Computer Vision*, 29-1:5–28, 1998.

[12] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101, 1998.

[13] J. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[14] S. LaValle, H. González-Baños, C. Becker, and J. Latombe. Motion strategies for maintaining visibility of a moving target. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 731–736, 1997.

[15] R. Murrieta, A. Sarmiento, and S. Hutchinson. A motion planning strategy to maintain visibility of a moving target at a fixed distance in a polygon. In *IEEE Int. Conf. on Robotics & Automation*, 2003.

[16] R. Murrieta-Cid, H. H. González-Baños, and B. Tovar. A reactive motion planner to maintain visibility of unpredictable targets. In *Proc. IEEE. Int. Conf. on Robotics & Automation*, pages 4242–4248, 2002.

[17] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *Proc. Int. Jnt. Conf. on Articial Intelligence*, page 477484, 2003.

[18] D. B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, AC-24(6):843854, December 1979.

[19] D. Schulz, W. Burgard, D. Fox, and A. Cremens. People tracking with mobile robots using sample-based joint probabilistic data association filters. *International Journal of Robotics Research (IJRR),*, 22(2):99–116, 2003.

[20] R. Smallwood and E. Sondik. The optimal control of partially observable markov processes over a nite horizon. *Operations Research*, 21:1071–1088, 1973.

[21] R. Vidal, O. Shakernia, H. Kim, D. Shim, and S. Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE Trans. on Robotics & Automation*, 18:662669, 2002.