Motion Planning for 3-D Target Tracking among Obstacles

Tirthankar Bandyopadhyay, Marcelo H. Ang Jr., and David Hsu

National University of Singapore, Singapore 117543, Singapore
{tirthankar,mpeangh}@nus.edu.sg, dyhsu@comp.nus.edu.sg

Summary. The goal of target tracking is to compute motion strategies for a robot equipped with visual sensors, so that it can effectively track a moving target despite obstruction by obstacles. It is an important problem with many applications in robotics. Existing work focuses mostly on the 2-D version of the problem, partly due to the complexity of dealing with 3-D visibility relationships. This paper proposes an online algorithm for 3-D target tracking among obstacles, using only local geometric information available to a robot's visual sensors. Key to this new algorithm is the definition and efficient computation of a risk function, which tries to capture a target's ability in escaping from the robot sensors' visibility region in both short and long terms. The robot then moves to minimize this risk function locally in a greedy fashion. In the absence of occlusion by obstacles, the standard tracking algorithm based on visual servo control can be considered a special case of our algorithm. Experiments show that the new algorithm generated interesting tracking behaviors in three dimensions and performed substantially better than visual servo control in simulation.

1 Introduction

Target tracking is an important task for autonomous robots. The goal of this work is to construct motion strategies for a robot equipped with visual sensors so that it can effectively track a moving target despite obstruction by obstacles. More precisely, the robot should always keep the target within the sensors' visibility region. Target tracking has attracted considerable attention in recent years [1, 4, 3, 8, 12, 13], but all the previous efforts have focused on the 2-D case. In this work, we build upon our earlier work on 2-D target tracking using local geometric information [1] and extend it to the 3-D case. This substantially expands the range of applications and potentially improves tracking performance.

Target tracking has many interesting applications in robotics. In surgery, robotic cameras can keep a patient's organ or tissue being operated on under constant observation, despite obstructions by people or instruments. In home environments, mobile or humanoid robots help to watch over the elderly or children. In wildlife monitoring, deep-sea underwater autonomous vehicles (UAVs) need to navigate in clutter environments while tracking marine species. The military and security offers many other potential applications. In all these applications, it is clearly advantageous to reason about visibility and plan motion in the 3-D rather than the 2-D space.

Moving from the 2-D to the 3-D space also offers opportunities to improve tracking performance. The 3-D space is more flexible: the robot gains one additional degree of freedom to maneuver, which potentially improves tracking performance. For example, a robot helicopter follows and monitors a ground target. If the target turns around at the corner of a building, the helicopter may choose to fly over the building to keep the target visible, instead of following the target to turn. However, the same flexibility leads to several challenges. Just as the robot, the target may also gain more room to maneuver and more easily escape from the robot sensors' visibility region. In addition, the visibility relationships in 3-D are more complex than those in 2-D.

Key to our algorithm is the definition of *risk*, which tries to capture the target's ability in escaping from the 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 comprise can be achieved, using only local information available to the robot's visual sensors. By analyzing the local geometry, our algorithm computes the global risk function as a weighted sum of components, each associated with a single visibility constraint. It then chooses an action so that the robot maneuvers to maintain all the visibility constraints against the target as well as possible in a weighted average sense.

Target tracking is a special class of motion planning problems that takes into account both *motion constraints* and *visibility constraints*. In classic motion planning [7], we consider only motion obstructions due to obstacles in the environments or the robot's mechanical construction, and the goal is to find a path for the robot to reach a specified destination, while avoiding the obstacles. In target tracking, we consider not only motion obstructions, but also visibility obstructions due to the obstacles, and the goal is to keep the target visible for as long as possible. Visibility relationships thus play a prominent role in deciding the robot's motion. The target tracking problem considered here is closely related to active sensing (*e.g.*, [10]). Related problems have been studied in various domains, *e.g.*, visual servo control [5], automatic target interception in missile control (*e.g.*, [15]), and visual tracking in computer vision (*e.g.*, [6]), but in these domains, one does not try to move the sensor actively in order to avoid motion or visibility obstructions caused by obstacles.

2 Related work

Many variants of the target tracking problem have been studied in the literature, but they focus mostly on the 2-D case [1, 4, 3, 8, 12, 13]. Suppose that both the environment and the target trajectory are unknown in advance, but the robot is equipped with visual sensors to gather information on the environment and the target. One approach is to move the robot so as to minimize an objective function that models the risk for the target to escape from the visibility region of the robot's sensor [4, 12, 1]. Our work follows this approach and extends it to three dimensions.

There has been little work on the 3-D tracking problem. One reason is that the 3-D visibility relationships are significantly more complex than their 2-D counterparts. Although there are data structures for maintaining visibility relationships globally,

e.g., aspect graphs [14] and visibility complexes [2], processing all the critical visibility events efficiently in a 3-D environment is a difficult task. The work of Lazebnik tries to characterize and process these visibility-events for a visibility-based pursuit-evasion problem [9]. In principle, it is possible to develop a tracking algorithm based on such a global visibility analysis, but an efficient algorithm has not yet appeared. The existing algorithms on 3-D tracking mostly rely on visual servo control [11, 16]. They have been used to control a team of unmanned aerial and ground vehicles for target tracking in a probabilistic game framework [16], but do not take into account the effect of visual occlusion by obstacles.

Just like its counterparts in 2-D [1, 4, 12], our 3-D tracking algorithm provides a good trade-off between tracking algorithms based on visual servo control and those based on dynamic programming [8, 3]. Like visual servo control, our algorithm uses only local information, but it tries to infer the long-term effects of the robot's actions approximately, using the risk function. Dynamic programming can reason about these long-term effects more accurately, but requires prior knowledge of the environment and the target behavior. Such information is often difficult to acquire in practice.

3 Problem formulation

The objective of the robot is to keep the target visible at all times. The tracking environment is a 3-D space cluttered with obstacles. Such environments may occur either indoors (*e.g.*, tracking a human in a regular home or office environment) or outdoors (*e.g.*, tracking a target in an urban environment).

The robot and the target are modeled as free flying point objects with no motion constraints, except for bounds on their maximum speeds, V and V', respectively. We assume $V' \leq V$; otherwise, the problem becomes uninteresting if the target tries to escape by moving faster than the robot. The robot motion is modeled as a simple discrete-time transition equation. If at time t, the robot at position $\mathbf{x}(t)$ moves with a velocity $\mathbf{v}(t)$, its position at time $t+1$ is given by

$$\mathbf{x}(t+1) = \mathbf{x}(t) + \mathbf{v}(t)\Delta t,$$

with $|\mathbf{v}(t)| \leq V$ for all t. So the feasible position of the robot in Δt is a sphere of radius $V\Delta t$ centered at $\mathbf{x}(t)$.

Both the environment and the target motion are unknown to the robot *a priori*, but we assume that the target is initially visible to the robot. The robot uses 3-D visual sensors, *e.g.*, cameras or laser range finders, for sensing the target and its surroundings. We assume omnidirectional sensing capabilities for the sensors. Visibility is modeled here as simple line of sight sensing, bounded by a maximal range D_{max}. Thus, in an open space, the robot's visibility region is a sphere of radius D_{max} with the center at $\mathbf{x}(t)$. Obstacles in the environment may obstruct visibility. Let $\mathcal{F}$ denote the space that is within the ball and is free of obstacles. The set of all points $\mathbf{q}$ within $\mathcal{F}$ visible to the robot defines the visibility region, $\mathcal{V}$, of the robot at position $\mathbf{x}$:

$$\mathcal{V} = \{\mathbf{q} \in \mathcal{F} | \overline{\mathbf{qx}} \subset \mathcal{F}\},$$

where $\overline{\mathbf{qx}}$ denotes the line segment joining $\mathbf{q}$ and $\mathbf{x}$. Clearly, $\mathcal{V}$ is star-shaped.

Now we can formalize the tracking problem as follows: *Find a sequence of actions—here, the robot velocities—such that at each instant in time, the target position lies inside* $\mathcal{V}$.

4 The tracking algorithm

The robot plans its actions using local geometric information from its sensors. The 3-D sensors have a visibility region, $\mathcal{V}$. Based on a polyhedral approximation for the environment geometry, $\mathcal{V}$ takes the shape of a generalized polyhedron bounded by two types of surfaces (shown in Fig.2*a*): the surfaces that are the boundaries of the polyhedral obstacles, *obstacle surfaces*, and the surfaces that lie in $\mathcal{F}$, *free surfaces*. The free surfaces can be further divided into *occlusion surfaces* that are caused by the obstruction of visibility and *range surfaces* that are caused by the visibility limits D_{max}. The free surfaces pose the risk of losing the target.

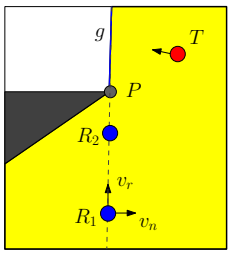


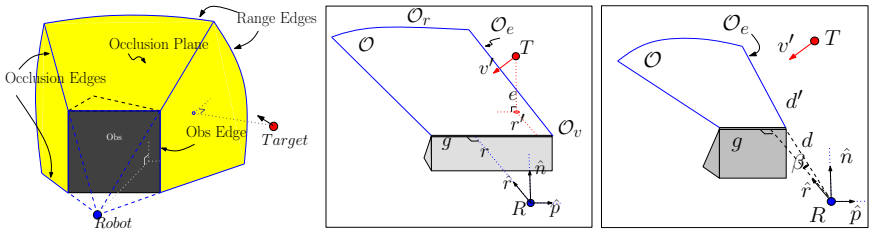
Fig. 1. Relative position determines risk

For successful tracking, the robot must balance the short-term goal of preventing the immediate loss of the target through these free surfaces 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 Fig.1. 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.

In the 3-D case, occlusion planes replace occlusion edges, and the robot and the target gain one additional degree of freedom to maneuver. However, the intuition on balancing the short-term and long-term goals, as illustrated in the 2-D example, remains basically the same. Below we propose a carefully constructed risk function to capture this intuition for each free surface. The total risk is a sum of these risks, weighted by the probabilities of the target's escaping through the corresponding surfaces. The robot's action is then a local greedy step to minimize the total risk.

4.1 Risk with respect to a Single Occlusion Surface

In general, the occlusion surfaces can be made up of a number of *occlusion planes* concatenated to each other. Adjacent occlusion planes meet in the *occlusion edges*.



(a) (b) (c)

Fig. 2. (a) 3-D Visibility Model (b) & (c) Parameters involved in the Risk Formulation

Let us take the occlusion plane $\mathcal{O}$ as shown in Fig.2*b*. $\mathcal{O}$ consists of the interior region ($\mathcal{O}_p$) bounded by a pair of occlusion edges ($\mathcal{O}_e$) at the lateral sides, obstacle edge (g) in front and range edge ($\mathcal{O}_r$) at the rear. Let us term the vertex at which $\mathcal{O}_e$ and g meet as *occlusion vertex* ($\mathcal{O}_v$). $\mathcal{O}$ depends on the robot's position w.r.t the obstacle, and so the robot can manipulate $\mathcal{O}$ by its motion. The robot motion normal to $\mathcal{O}$, along $\hat{\mathbf{n}}$, increases the plane's distance to the target by swinging about g. Similarly, motion parallel to g, along $\hat{\mathbf{p}}$, increases the distance of the $\mathcal{O}_e$ to the target by swinging $\mathcal{O}_e$ laterally about $\mathcal{O}_v$. Both these velocity components help in achieving the short term goal of preventing the target's immediate escape. On the other hand, motion towards $\mathcal{O}$, along $\hat{\mathbf{r}}$, improves the future tracking capability of the robot by moving closer to the obstacle. Introducing a reference frame on the robot aligned to these individual motion directions : $\hat{\mathbf{n}}$, $\hat{\mathbf{p}}$ and $\hat{\mathbf{r}}$, shown in Fig.2*b* & 2*c*, simplifies the analysis.

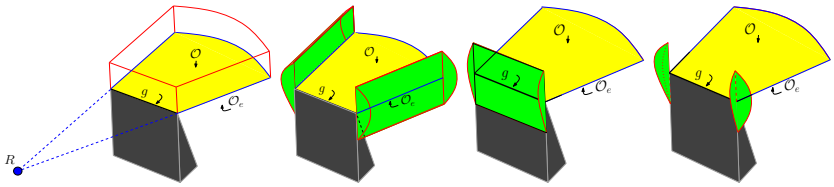
Let us now derive the risk formulation. The risk of the target's escape through $\mathcal{O}$ depends on its shortest distance of escape (SDE) to the plane, as denoted by e in Fig.2*b*. The smaller the e is, the higher is the risk of the target's escaping. The risk also depends on how well the robot can manipulate $\mathcal{O}$ away from the target. The effectiveness of manipulation depends on the relative positioning of the robot and the target w.r.t $\mathcal{O}$. Let r be the distance between the robot and g, and r' be the distance between the target and g, projected into the occlusion plane, shown in Fig.2*b*. If $r' > r$, the robot has an advantage in swinging $\mathcal{O}$ farther away from the target given the same velocity components of the robot and target normal to $\mathcal{O}$. Similarly in Fig.2*c*, if d' is the distance of the target's projection on $\mathcal{O}_e$ from $\mathcal{O}_v$ and d the corresponding measure for the robot, $d' > d$ gives the robot an advantage in swinging $\mathcal{O}_e$ about $\mathcal{O}_v$. This "relative vantage" in the robot's manipulation capabilities of $\mathcal{O}$ over the target can be encoded by a measure of spatial proximity to the occlusion.

Let us construct a vantage zone ($\mathcal{D}$), such that for any position of the target within that region, the robot cannot eliminate $\mathcal{O}$ (by moving towards $\mathcal{O}_e$), before the target reaches $\mathcal{O}$, given the target's and robot's current velocities. Mathematically,

$$\mathcal{D} = \{\mathbf{q} : \mathbf{q} \in \mathcal{V} \bigwedge (e(\mathbf{q}) < D_{safe})\} \tag{1}$$

where $e(\mathbf{q})$ is the shortest distance to escape for the target from $\mathbf{q}$ through $\mathcal{O}$ and D_{safe} is the minimum distance $\mathcal{O}$ must be kept from the target for the robot to be able to successfully prevent the target's escape.

This approach is similar to that proposed for 2-D tracking [1]. But the additional dimension in 3-D increases the complexity as now the occlusion plane can also be

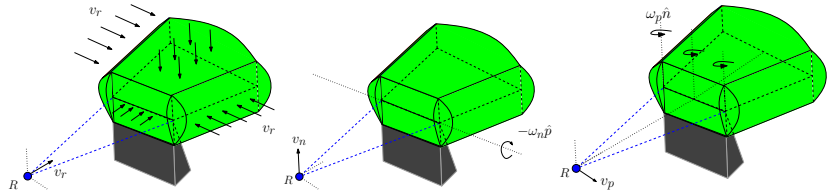


(*a*) $\mathcal{D}_N$ (*b*) $\mathcal{D}_L$ (*c*) $\mathcal{D}_R$ (*d*) $\mathcal{D}_V$

Fig. 3. Vantage Zone $\mathcal{D}$ for a single occlusion plane

shifted laterally along g in addition to the normal swing about g. Based on the proximity to the occlusion's features, $\mathcal{D}$ can be partitioned into four regions, shown in Fig.3.

- $\mathcal{D}_N$: The region in $\mathcal{D}$ that is closest to the interior of $\mathcal{O}$ i.e. $\mathcal{O}_p$. This region is depicted in Fig.3*a*.
- $\mathcal{D}_L$: The region in $\mathcal{D}$ nearest to the occlusion edge $\mathcal{O}_e$. $\mathcal{D}_L$ is depicted by semi-circular cylindrical pieces abutting the occlusion edges, Fig.3*b*.
- $\mathcal{D}_R$: The region in $\mathcal{D}$ closest to the obstacle edge g, as shown in Fig.3*c*. $\mathcal{D}_R$ is a semi-circular cylindrical piece abutting g.
- $\mathcal{D}_V$: The regions in $\mathcal{D}$ nearest to the occlusion vertex, $\mathcal{O}_v$, as shown in Fig.3*d*. $\mathcal{D}_V$ is a pair of spherical sectors.



(*a*) v_r (*b*) v_n (*c*) v_p

Fig. 4. The effect of robot velocities v_r,v_n and v_p on $\mathcal{D}$

As with $\mathcal{O}$, the robot can manipulate $\mathcal{D}$ by its motion. v_r causes $\mathcal{D}$ to shrink towards $\mathcal{O}$ (Fig.4*a*), v_n causes $\mathcal{D}$ to swing about g (Fig.4*b*), in the normal direction to the $\mathcal{O}$ by an angular velocity ω_n and v_p shears $\mathcal{D}$ along g by swinging $\mathcal{O}_e$ laterally by an angular velocity ω_p (Fig.4*c*). From Fig.2*b* & 2*c*, $\omega_n = -(v_n/r)\hat{\mathbf{p}}$, and $\omega_p = ((v_p \cos\beta - v_r \sin\beta)/d)\hat{\mathbf{n}}$.

In order to maintain a good relative vantage over the target, the robot must manipulate $\mathcal{D}$ such that the target is pushed out of $\mathcal{D}$ if it is inside, else move $\mathcal{D}$ as far from the target as possible. A good estimate of the relative vantage, is the shortest amount of time, $t_{\mathcal{D}}$, the target needs to reach the boundary of $\mathcal{D}$. $t_{\mathcal{D}}$ is positive if the target is inside $\mathcal{D}$ and negative if outside. We call $t_{\mathcal{D}}$ the *vantage time*. Let $\mathcal{P}$ be the point through which the target exits $\mathcal{D}$. $t_{\mathcal{D}}$ can then be computed by,

$$t_{\mathcal{D}} = \frac{dist(target, \mathcal{P})}{v_{eff}} \tag{2}$$

where v_{eff} is the relative velocity of the target w.r.t $\mathcal{P}$. Depending on the robot's motion, $\mathcal{P}$ can lie either in $\mathcal{D}_N$, $\mathcal{D}_L$, $\mathcal{D}_R$ or $\mathcal{D}_V$. Optimizing $t_{\mathcal{D}}$ over all the four regions involves finding an optimal velocity $v^\star$ such that, $v^\star = argmin(t_{\mathcal{D}}) \quad \forall \mathcal{P} \in \mathcal{D}_N, \mathcal{D}_L, \mathcal{D}_R, \mathcal{D}_V$. To simplify computation, let us approximate $t_{\mathcal{D}}$ by its lowest bound,

$$t_{\mathcal{D}} = min(t_{\mathcal{DN}}, t_{\mathcal{DL}}, t_{\mathcal{DR}}, t_{\mathcal{DV}}) \tag{3}$$

The optimization then reduces to finding $v^\star = \nabla t_{\mathcal{D}}$.

In order to find the analytical expressions for $t_{\mathcal{D}}$ in the four regions, let us take $V = V'$. This does not take away the generality of the derivation as we can introduce a scale factor η such that, $V = \eta V'$. For the following, we take $\eta = 1$, which gives $D_{safe} = r$.

Computing $t_{\mathcal{DN}}$

Fig.5 shows a piece of $\mathcal{D}_N$. The effective velocity of the the planar $\mathcal{D}_N$ surface w.r.t the target is $v_{eff} = v_r + \omega_n r' - v'_n$ and effective $dist(target, \mathcal{P}) = D_{safe} - e$ giving,

$$t_{\mathcal{DN}} = \frac{r - e}{v_r + \omega_n r' - v'_n}$$

Interestingly, the result takes the same form as risk optimization in 2-D [1].

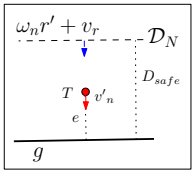


Fig. 5. Computing $t_{\mathcal{DN}}$

Computing $t_{\mathcal{DL}}$

Let us ignore v_r for now. Fig.6 shows a piece of $\mathcal{D}_L$. The effective motion of the target and $\mathcal{D}_L$ can be shown to lie along section AA'. We omit the proof for brevity. Let us introduce a coordinate system fixed at occlusion vertex $\mathcal{O}_v$ $\{\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}}\}$. The target position $\mathbf{t}$ w.r.t the new coordinates is $\mathbf{t} = -t_x\hat{\mathbf{i}} + d'\hat{\mathbf{j}} + e\hat{\mathbf{k}}$. The effective velocity, v_{eff}, is a resultant of three components: the normal and lateral swing, the shrinking of $\mathcal{D}$ and the velocity of the target v'. $v_{eff} = \omega_n \hat{\mathbf{p}} \times \mathbf{t} + \omega_p \hat{\mathbf{k}} \times \mathbf{t} - v'$. The effect of v_r is to shrink $\mathcal{R}$ towards $\mathcal{O}_e$. This gives the condition, $\mid \mathbf{t} + v_{eff} t_{\mathcal{D}} \mid_{ik} = D_{safe} - v_r t_{\mathcal{D}}$. Solving for $t_{\mathcal{DL}}$ gives,

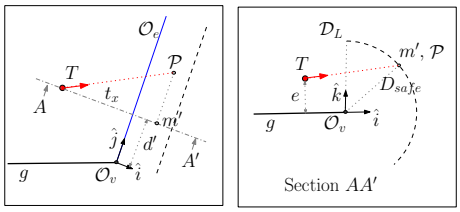


Fig. 6. Computing $t_{\mathcal{DL}}$

$$t_{\mathcal{DL}} = \frac{-(-t_x v'_x + e v'_z + r v_r) \pm \sqrt{(-t_x v'_x + e v'_z + r v_r)^2 - 4({v'_x}^2 + {v'_z}^2 - {v_r}^2)({t_x}^2 + e^2 - r^2)}}{2({v'_x}^2 + {v'_z}^2 - {v_r}^2)}$$

Computing $t_{\mathcal{DR}}$ and $t_{\mathcal{DV}}$

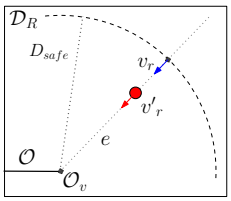


Fig. 7. Computing $t_{\mathcal{DR}}$

$\mathcal{D}_R$ and $\mathcal{D}_V$ behave in a similar way that, neither ω_n nor ω_p cause any relative velocity of the surface towards the target in this case, as shown in Fig.7. Taking the radial component of velocity and distance, $v_{eff} = v_r - v'_r$ and $dist(target, \mathcal{P}) = D_{safe} - e$ giving,

$$t_{\mathcal{DR}} = t_{\mathcal{DV}} = \frac{r - e}{v_r - v'_r}$$

$t_{\mathcal{DV}}$ is given by the same expression for its corresponding case.

4.2 Risk with respect to a Single Range Surface

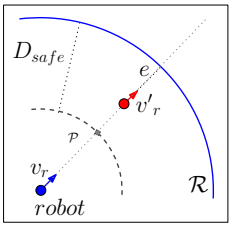


Fig. 8. Computing $t_{\mathcal{D}}$ for Range

The robot cannot eliminate range surfaces ($\mathcal{R}$) by moving towards it. In the definition of $\mathcal{D}$, we re-define D_{safe} based on just the radial component of the robot and the target, as shown in Fig.8. Then the effective velocity becomes $v_{eff} = v_r - v'_r$, and $dist(target, \mathcal{P}) = D_{safe} - e$. e is then calculated towards the nearest point in the $\mathcal{R}$. The expression for $t_{\mathcal{D}}$ is the same as that of $t_{\mathcal{DR}}$.

An interesting thing to note is that the behavior generated by the range surfaces alone, makes the robot move towards the target. This is exactly the visual servo behavior. This shows that visual servo is a special case of vantage tracking when there are no occlusions.

Once $t_{\mathcal{D}}$ is found, the gradient of $t_{\mathcal{D}}$ computed at the current values gives the optimal direction to move.

$$\nabla t_{\mathcal{D}} = \frac{\partial t_{\mathcal{D}}}{\partial v_r}\hat{\mathbf{r}} + \frac{\partial t_{\mathcal{D}}}{\partial v_n}\hat{\mathbf{n}} + \frac{\partial t_{\mathcal{D}}}{\partial v_p}\hat{\mathbf{p}} \tag{4}$$

Multiple occlusion planes are handled by weighing the individual actions by the probability of the target's escape through the particular occlusion plane,

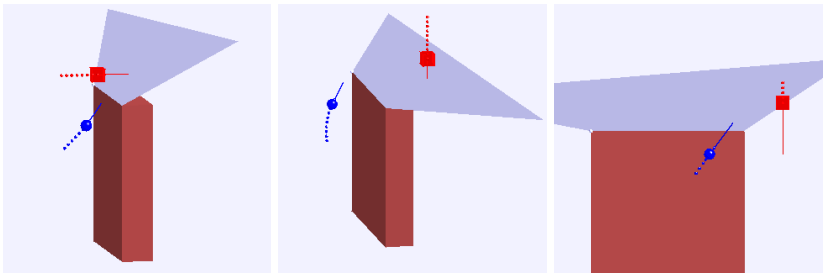
$$\nabla \Phi = \sum_i p_i \left(\frac{\partial t_{\mathcal{D}}}{\partial v_r}\hat{\mathbf{r}} + \frac{\partial t_{\mathcal{D}}}{\partial v_n}\hat{\mathbf{n}} + \frac{\partial t_{\mathcal{D}}}{\partial v_p}\hat{\mathbf{p}}\right).$$

4.3 Escape Probability for the Target

For finding the probability of the target's escape through any particular occlusion p_i, we predict the motion of the target by independent distributions $p(\theta)$ and $p(\phi)$ on its azimuth (θ) and zenith (ϕ) angle. We assume that the target speed remains constant, but in general, this can also be modeled by a distribution $p(s)$. The probability is measured in terms of the integral of the volume subtended by the plane,

$$p_i(\phi, \theta) = \int_{\theta_1}^{\theta_2} \int_{\phi_1}^{\phi_2} p(\phi)p(\theta)d\phi d\theta$$

For lack of space we do not go into the details of solving this double integral.



(a) CASE I (b) CASE II (c) CASE III

Fig. 9. Control Experiments

5 Experiments

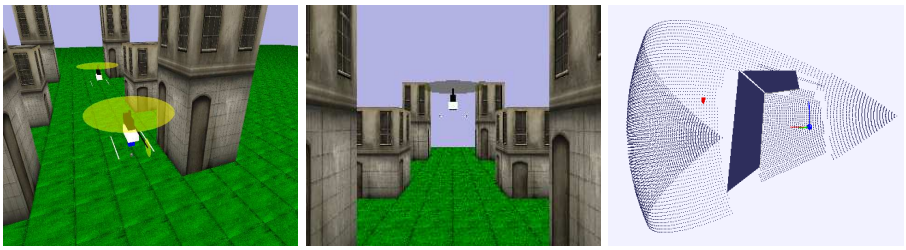
We performed different tests on the algorithm by providing it with various environments and analyzing the results. In the first test, we check for specific behaviors of the algorithm with different configurations of the target in a simple environment, Fig.9. In the second test, we run the algorithm in a more complex urban environment amid sensor uncertainties, Fig.10, and analyze its performance metrics.

We create a simple scenario in Fig.9, where the target (red cube) tries to escape the visibility of the robot (blue sphere), by moving behind the obstacle (maroon wall). To analyze the fundamental characteristics of the robot motion, in response to the target's motion against an occlusion plane, we turn off all the occlusion planes except the blue plane at the top of the wall. The dotted lines depict the path executed by the target and the robot, while the solid segments show their current heading. For all the cases, the robot is placed in front of the wall. The target's path is unknown to the robot. The robot's velocity is generated at each step by Eqn.3 & 4.

CASE I : When the target is placed in front of the wall, shown in Fig.9*a*, its shortest path of escape from the robot's visibility passes through the top edge of the wall. This means that any amount of swinging of the occlusion plane by the robot would be fruitless and the robot should move towards this edge. This behavior is reproduced by the robot, as v_r is the only component produced by Eqn.3 & 4.

CASE II : Next, let the target be placed above the wall somewhere middle along its horizontal length, shown in Fig.9*b*. For such a position, the target's closest point of escape is its normal projection on the occlusion plane. In such a situation then, it makes sense to swing the plane away from the target. The algorithm manages to produce a combination of v_n and v_r to address the scenario. v_n helps in swinging the plane, and v_r helps in improving the vantage by moving closer to g. This combination, that balances the long term and short term goals, generates a curved path as seen in the figure.

CASE III : If the target is placed not at the middle, but towards one end over the wall, a lateral swing can increase the shortest distance value in addition to the normal swinging motion. This is characteristic to 3-D environments where the robot can prevent the target's escape by shifting the plane laterally away from under the target. In general, people tend to show this kind of behavior in such a



(a) (b) (c)

Fig. 10. Realistic simulation setup using Gazebo. (a) Environment setup, (b) Robot viewpoint, (c) Extracting $\mathcal{O}$ from 3-D range scan

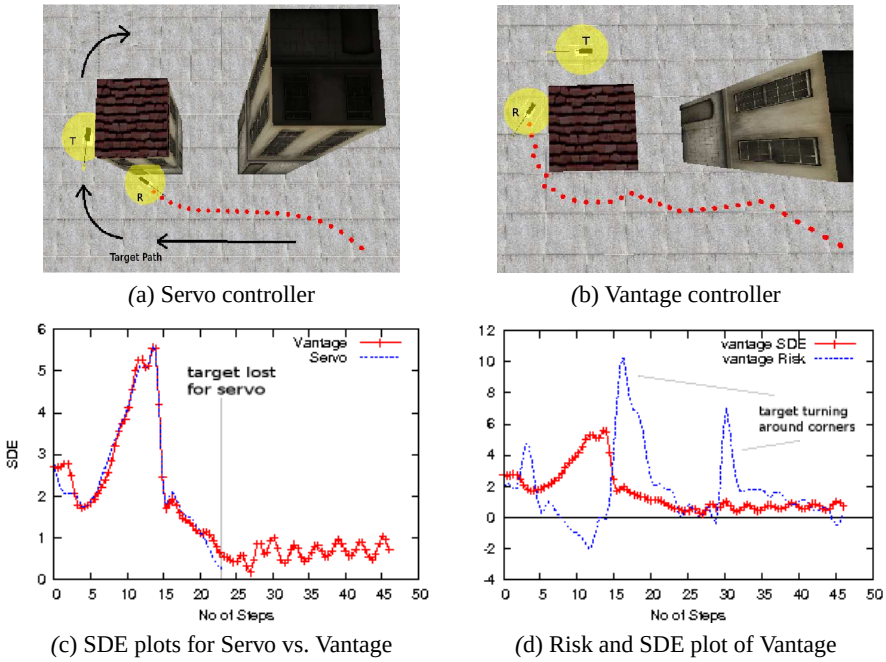
situation. This behavior is also shown by the algorithm as it generates a horizontal component v_p, in addition to v_n and v_p. This shows that the algorithm is able to exploit the additional dimension available to the robot in 3-D. Moreover, this is achieved using only local information.

We next test the effectiveness of our algorithm in a realistic scenario by implementing it in the Gazebo robot simulator [17]. Gazebo is a multi-robot simulator for both indoor and outdoor environments in 3-D. It generates realistic sensor feedback, object collision, and dynamics. Using Gazebo, we build an urban environment in which a robot helicopter tracks our target, another helicopter, shown in Fig.10. The environment has buildings of various sizes, separated by alleys and pathways. We mount a 3-D sweeping laser range finder on the robot helicopter. In general, this can be replaced by any reliable vision system without much impact on our algorithm. The 3-D range data is processed to extract occlusion planes. Fig.10*c* shows an example. The set of points is the sensor data from the laser range finder. The dark blue planes are the occlusion planes obtained by the range discontinuity upon thresholding. The red dot indicates the target position.

We compare our algorithm with the popular visual servo algorithm. Both are online algorithms that can be implemented using exactly the same setup. To evaluate the performance, we compare the shortest distance to escape (SDE) from the target position to the nearest occlusion plane. Clearly, if an algorithm alway maintains superior SDE throughout, thereby keeping the target away from the possible escape regions, it can be considered to have a better tracking performance.

Figures 11*a* & 11*b*, show the tracking results for the servo algorithm and our vantage algorithm, respectively. The target executes an identical path, which is marked in Fig.11*a*. The robot starts from the same position in the lower right part of the figures. The dotted paths show the robot's paths under the control of the two algorithms.

The servo tracker loses the target at step 23, whereas the vantage tracker continues until we stop the simulation at step 46, at which time the target is still visible. The SDE plots, in Fig.11*c*, show that the two trackers have comparable performance until around step 20. After that, the SDE for the servo tracker drops to 0, while the vantage tracker still maintains good SDE values and is able to continue tracking the target. The reason behind the success of the vantage tracker becomes clearer when we look at Fig.11*d*, which plots the risk values computed by the vantage tracker at



(a) Servo controller (b) Vantage controller

(c) SDE plots for Servo vs. Vantage (d) Risk and SDE plot of Vantage

Fig. 11. Experimental Results

each step. We see that the risk values peak for certain steps. Careful inspection reveals that such peaks occur whenever the target turns around a corner. For example, the target turns right sharply in steps 15–20, then again around step 30, and we have peaks in the risk plot accordingly. From the peaked risk values, the vantage tracker perceives the danger of losing the target in the near future and moves to reduce it, thereby successfully keeping the target visible. The servo tracker does not consider the effect of occlusion by obstacles and loses the target.

6 Conclusion

This paper proposes an online algorithm for 3-D target tracking among obstacles. It uses a carefully defined risk function to reason about robot actions in order to balance between short and long term benefits. As the algorithm uses only local geometric information available to the robot's 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, because the robot's action is computed using sensor data acquired in the current step only. This improves the reliability of tracking. Simulation results show that the new algorithm generated interesting tracking behaviors in 3-D and performed substantially better than an algorithm based on visual servo control.

Currently, we are extending the algorithm for more realistic sensor models, in particular, by incorporating field of view (FoV) constraints. Like range constraints, FoV constraints are different in nature from occlusion constraints, because they result

from the sensor limitations and cannot be eliminated through robot motion. FoV constraints are modeled as pseudo occlusion surfaces, and the risk is calculated based on the time of escape of the target rather than the vantage time. We are also improving the robot motion models by taking into account the kinematics and dynamics of the robot. For example, non-holonomic motion limits the space of feasible velocities available to the robot and can be incorporated as constraints when the robot chooses its action. Our implementation of visibility region extraction from 3-D laser data can also be significantly improved by using advanced techniques from computer vision.

References

1. 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.
2. F. Durand, G. Drettakis, and C. Puech. The 3d visibility complex. In *ACM Trans. on Graphics*, volume 21(2), pages 176–206. ACM Press, April 2002.
3. 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.
4. 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.
5. S. Hutchinson, G. Hager, and P. Corke. A tutorial on visual servo control. *IEEE Trans. on Robotics & Automation*, 12(5):651–670, 1996.
6. D. Huttenlocher, J. Noh, and W. Rucklidge. Tracking non-rigid objects in complex scenes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 93–101, 1993.
7. J. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.
8. 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.
9. S. Lazebnik. Visibility-based pursuit-evasion in three-dimensional environments. *Beckman CVR TR*, 2001.
10. J. Maver and R. Bajcsy. Occlusions as a guide for planning the next view. *IEEE Trans. on Pattern Analysis & Machine Intelligence*, 15(5):417–433, 1993.
11. L. O. Mejias, S. Saripalli, P. Cervera, and G. S. Sukhatme. Visual servoing of an autonomous helicopter in urban areas using feature tracking. *Journal of Field Robotics*, 23(3):185–199, 2006.
12. R. Murrieta-Cid, 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.
13. R. Murrieta-Cid and S. Hutchinson. Surveillance strategies for a pursuer with finite sensor range. *Int. J. Robotics Research*, 26(3):233–253, 2007.
14. H. Plantinga and C. Dyer. Visibility, occlusion, and the aspect graph. *Int. J. Computer Vision*, 5(2):137–160, 1990.
15. T. Song and T. Um. Practical guidance for homing missiles iwth bearings-only measurements. *IEEE Tran. on Aerospace & Electronic Systems*, 32:434–443, 1996.
16. R. Vidal, O. Shakernia, H. Kim, D. Shim, and S. Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE Trans. on Robotics and Automation*, 18(5):662 – 669, Oct. 2002.
17. *http://playerstage.sourceforge.net*.