Collision Prediction Among Polygons with Arbitrary Shape and Unknown Motion

Yanyan Lu, Zhonghua Xi and Jyh-Ming Lien

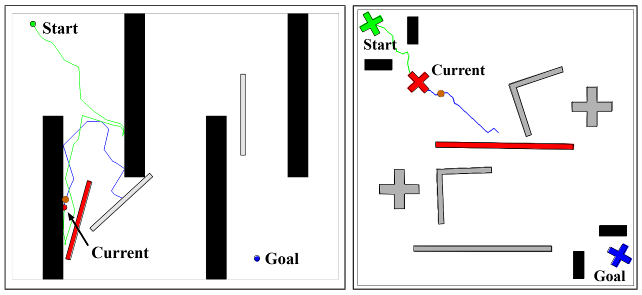
***Abstract*— Collision prediction is a fundamental operation for planning motion in dynamic environment. Existing methods usually exploit complex behavior models or use dynamic constraints in collision prediction. However, these methods all assume simple geometries, such as disc, which significantly limit their applicability. This paper proposes a new approach that advances collision prediction beyond disc robots and handles arbitrary polygons. Our new tool predicts collision by assuming that obstacles are *adversarial*. Comparing to an online motion planner that replans periodically at *fixed time interval* and planner that approximates obstacle with discs, our experimental results provide strong evidences that the new method significantly reduces the number of replans while maintaining higher success rate of finding a valid path. Our geometric-based collision prediction method provides a tool to handle highly complex shapes and provides a complimentary approach to those methods that consider behavior and dynamic constraints of objects with simple shapes.**

I. Introduction

Imagine a scenario where a robot navigates itself through a disaster zone filled with static obstacles, mobile robots carrying debris with various sizes and shapes and mobile manipulators picking up and loading debris on top of the mobile robots or conveyor belts. In this scenario, the robot must plan its path without knowing how the other robots will move, and, in practice, it is inefficient to consider the motion of other robots before planning its own movement. Similarly, a robot requires the same ability to navigate through a factory, warehouse or airport among other mobile robots manipulating and carrying commercial goods with various sizes and shapes. Fig. 1 illustrates two of such examples where a mobile robot, which is modeled either as a point or a polygon, navigates through environments filled with static and dynamic obstacles with arbitrary shapes. In motion planning literature, this problem is usually known as *online motion planning* or *sensor-based motion planning*.

Online motion planning methods usually exploit the idea of temporal coherence to gain better efficiency by repairing the invalid portion of the path or (tree-based or graph-based) roadmaps [1], [2], [3], [4] since the changes in the configuration space is usually small from frame to frame. These planning strategies are often known as *replanning methods* [5], [6], [7], [8], [9]. Although these replanning methods are efficient, almost all existing frameworks update the environmental map and then replan periodically *at fixed time interval*. That is, even if there are no changes in the

All authors are with the Department of Computer Science, George Mason University, USA, 22030. This work is supported in part by NSF IIS-096053, EFRI-1240459, CNS-1205260, FHWA DTFH61-10-R-00036, AFOSR FA9550-12-1-0238 and USGS.



(a) Point robot (b) Polygonal robot

Fig. 1. Two examples of a mobile robot moving from the upper-left corner to the lower-right corner through environments filled with static (black) and dynamic (grey) obstacles whose motion is unknown to the robot. Bounding these moving obstacles with circles can lead to poor collision prediction and result in many unnecessary replanning. Our method predicts the collision time for obstacles with arbitrary shapes. The obstacles shown in red are the ones with the earliest collision times with respect to the current configurations of the robot (also shown in red).

configuration space, motion planner will still be invoked to replan. The situation is even worse when replanning is not done frequently enough: Paths that are believed to be valid may become unsafe.

Motivated by this issue, several strategies [10], [11], [12], [13], [8], [14], [15] have been proposed to replan *adaptively* only at the critical moments when the robot and obstacles may collide. These critical moments are usually detected by *collision prediction* methods. The main challenge in predicting collision steams from the assumption that obstacle's motion is unknown. Existing methods in collision prediction exploit complex behavior prediction [14], [15] or consider dynamic constraints [10], [11], [13], [16]. However, these methods all assume either translational or disc objects, which significantly limit their applicability. This is particularly true in cluttered environments, such as those in the aforementioned applications and the examples in Fig. 1, where moving obstacles can have arbitrary shapes and sizes. As we will show later, bounding these moving obstacles with discs can lead to arbitrarily poor collision estimation.

In this paper, we propose a new geometric tool that advances collision prediction beyond the translational and disc objects and can handle arbitrary polygons. The basic framework introduced in this paper models the obstacles as *adversarial agents* that will minimize the time that the robot remains collision free. As a result, a robot can actively determine its next replanning time by conservatively estimating the amount of time (i.e., *earliest collision time*) that it can stay on the planned path without colliding with the obstacles. The idea of earliest collision time and conservative advancement are detailed in Sections III to VI. This prediction is determined only based on the last known positions of the obstacles and their maximum linear and angular velocities. In our experimental results (in Section VIII), we demonstrate that an online planner using the proposed collision prediction method significantly reduces the number of replannings while maintaining the same or higher success rate of finding a valid path than (1) planner that replans periodically at fixed time intervals and (2) planner that bounds obstacles with circles. In essence, our main contribution is a geometric-based collision prediction method that can handle highly complex shapes. This tool provides a complimentary approach to the methods that consider complex behavior prediction or handle dynamic constraints but with only simple shapes.

II. Related Work

Motion planning problems involving dynamic environments can be roughly classified into two categories: (1) The trajectory of every moving obstacle is fully known in advance, and (2) the trajectory of a moving obstacle is partially or completely unpredictable. Since our work falls into the second category, we will focus on reviewing recent works considering unknown environments.

Collision Avoidance. Due to little knowledge of the environment, safety becomes very important and challenging in path planning in unknown environments [17], [18], [12], [19], [20], [21], [22], [23], [24], [25]. Fraichard and Asama [21] provided the formal definitions of two new concepts: inevitable collision state (ICS) and inevitable collision obstacle (ICO). If the robot is in an ICS, no matter what its future trajectory is, a collision eventually occurs with an obstacle in the environment. ICO is a set of ICS yielding a collision with a particular obstacle. Shiller et al. [19] proposed a motion planner based on Velocity Obstacles (VO) for static or dynamic environments. The time horizon for a velocity obstacle is computed based on the current positions of robot and the obstacle as well as control constraints. With this adaptive time horizon strategy, the velocity obstacle tightly approximates the set of ICS. Gomez and Fraichard [22] proposed another ICS-based collision avoidance strategy called ICS-AVOID. ICS-AVOID aims at taking the robot from one non-ICS state to another. The concept of Safe Control Kernel is introduced and it guarantees ICS-AVOID can find a collision-free trajectory if one exists. Recently, Bautin et al. [26] proposed two ICS-checking algorithms. Both algorithms take a probabilistic model of the future as input which assigns a probability measure to the obstacles' future trajectories. Instead of answering whether a given state is an ICS or not, it returns the *probability* of a state being an ICS. Wu and How [16] extended VO to moving obstacles with constrained dynamics but move unpredictably.

The work closes to the spirit of our new method is by van den Berg and Overmars [8]. Their work assumes that the robot and all obstacles are discs and it conservatively models the swept volume of an obstacle over time as a cone with the slope being its maximum velocity. Therefore, no matter how the obstacle moves, it is always contained in the cone. However, these assumptions can be unrealistic for many applications. For arbitrary shapes, computing the swept volumes is nontrivial.

Collision Prediction. Since the robot has partial or no information about the environment, it is very difficult to plan a collision free path for it to move through a field of static or dynamic obstacles to a goal. One of the biggest challenge is to predict possible collisions with dynamic obstacles whose trajectories are unknown. There exists a lot of work which checks collisions at a sequence of fixed time steps [7], [27], [28], [29], [30]. For example, van den Berg et al. [7] performed collision detections at fixed time intervals (every 0.1 seconds in their experiments). Both the robot and dynamic obstacles were modeled as discs moving in the plane. Moreover, the future motions of a moving obstacle were assumed to be the same as its current motions. In order not to miss any collisions, they either increased the number of time steps or assumed the objects move very slowly.

There are also works which adaptively changed the frequency of collision checks: collisions are more frequently checked for two objects which are more likely to collide. Hayward et al. [10], Kim et al. [13] and Hubbard [11] assumed that the maximum magnitude of the acceleration is provided for each object. Hayward et al. calculated the amount of time within which two moving spheres are guaranteed not to collide with each other. Then more attention was adaptively paid to objects which are very likely to collide. Hubbard first detected collisions between the bounding spheres of two objects. Then the pairs of objects whose bounding spheres intersect are further checked for collisions using sphere trees that represent the objects. Kim et al. [13] first computed the *time-varying bound volume* for each moving sphere with its initial position, velocity and the maximum magnitude of its acceleration. As time goes by, the radius of this *time-varying bound volume* increases and it is guaranteed to contain the sphere at any time in the future. For two moving spheres, whenever their *time-varying bound volume*s intersect, they are checked for actual collision. Chakravarthy and Ghose [12] proposed *collision cone* approach (similar as velocity obstacle) for predicting collisions between any two irregularly shaped polygons translating on unknown trajectories. All these methods are limited to discs, spheres or translational objects. Our new tool allows polygons with arbitrary shape (even non-simple polygons) with rotation.

Almost all existing works collect sensory data and update its environmental information at fixed times. As a result, either updating is redundant or the situation is even worse if update is performed not frequently. The robot may be at some state which leads it to be in unavoidable collisions. To address this, we propose to update environmental belief when necessary by exploring temporal coherence of obstacles and predict a critical time t such that the robot is guaranteed to move safely along its current path until t.

III. Overview of Our Method

Planning a path in environments populated with obstacles with unknown trajectories usually involves two steps: (1) find an initial path Π based on known information and then (2) modify Π as the robot receives new information from its onboard sensors at *fixed times*. To provide a more concrete framework for our discussion, we assume that the robot $\mathbf{R}$ still plans a path Π based on its current belief of the state of the workspace. However, instead of determining if Π is still safe to traverse at fixed time, $\mathbf{R}$ determines the critical moment t that Π may become invalid. The robot budgets a certain amount of time $\triangle t$ before this critical moment t to update its belief and replan if necessary. We would like to emphasize that this setting is merely a framework among other applications of collision prediction.

Because the trajectory of the obstacles in workspace is unknown, the critical moment t can only be approximated. To ensure the safety of the robot, our goal is to obtain conservative estimation $t' \leq t$ of the unknown value t. Follow the naming tradition in collision detection, we call such an estimation *conservative advancement* on Π and denote it as CA_Π. To compute CA_Π, the robot assumes that all obstacles are adversarial. That is, these adversarial obstacles will move in order to minimize the time that Π remains valid.

Contrary to traditional motion planning methods, the calculation of CA_Π (performed by the robot) in some sense reverses the roles of robot and obstacles. The robot $\mathbf{R}$ is now fixed to the path Π, thus the configuration of $\mathbf{R}$ at any given time is known. On the other hand, the obstacles' trajectories are unknown but will be planned to collide with $\mathbf{R}$ in the shortest possible time. As we will see later, the motion strategy for an obstacle $\mathbf{O}_i$ will only depend on the the maximum translational velocity v_i and a maximum angular velocity ω_i around a given reference point o, where o is the rotation center specified by users for an obstacle.

A. Estimate Conservative Advancement on Path Π

Without loss of generality, the problem of estimating CA_Π can be greatly simplified if we focus on only a single obstacle and a segment of path Π. Let Π be a sequence of free configurations $\Pi = \{c_1, c_2, ..., c_n\}$ with $c_1 = \mathbf{S}$ and $c_n = \mathbf{G}$, where the $\mathbf{S}$ and $\mathbf{G}$ are start and goal configurations, respectively. Given a segment $\overline{c_j c_{j+1}} \subset \Pi$, we let $ECT_{i,j}$ be the earliest collision time (ECT) that $\mathbf{O}_i$ takes to collide with the robot on $\overline{c_j c_{j+1}}$. Then we have $CA_\Pi = \min_i(\min_j(ECT_{i,j}))$, where $1 \leq i \leq |\mathbf{O}|$ and $1 \leq j < n$. Note that $ECT_{i,j}$ is infinitely large, if $\mathbf{O}_i$ cannot collide with $\mathbf{R}$ before $\mathbf{R}$ leaves $\overline{c_j c_{j+1}}$.

Lemma 3.1: If $ECT_{i,j} \neq \infty$, then $ECT_{i,j} \leq ECT_{i,k}$, $\forall k > j$

That is once an earliest collision time is detected for a path segment $\overline{c_j c_{j+1}}$, it is not necessary to check all its subsequent segments $\overline{c_k c_{k+1}}$ with $j < k < n$. In Section III-B, we will provide an overview on how $ECT_{i,j}$ can be computed.

Before we proceed our discussion, we would like to point out that our method does not consider collisions between the obstacles. Although this makes our estimate more conservative, the obstacle with the earliest collision time rarely collides with other obstacles.

B. Earliest Collision Time (ECT)

Given a segment $\overline{c_j c_{j+1}} \subset \Pi$ of path in C-space, our goal is to compute the earliest collision time $ECT_{i,j}$ when obstacle $\mathbf{O}_i$ hits robot $\mathbf{R}$ somewhere on $\overline{c_j c_{j+1}}$. Assume $\mathbf{R}$ starts to move on Π at time 0.

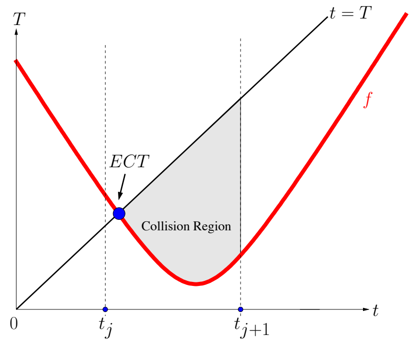


Fig. 2. The red (thicker) curves in both figures are plots of the earliest arrival time $f(t)$ for an obstacle. Black straight lines are plots of $g(t) : t = T$. When there is at least one intersection (blue dot) between $f(t)$ and $g(t)$, **collision region** is not empty.

Since the robot $\mathbf{R}$ moves along a known path Π, $\mathbf{R}$ knows when it reaches any configuration $c \in \Pi$. Let t be the time that $\mathbf{R}$ takes to reach a configuration $c(t) \in \overline{c_j c_{j+1}}$ and let T be the time when $\mathbf{O}_i$ reaches this $c(t)$. Because $\mathbf{O}_i$ is constrained by it maximum linear and angular velocities v_i and ω_i, there must exist an earliest time $\hat{T}$ for $\mathbf{O}_i$ to reach any $c \in \overline{c_1 c_2}$ without violating these constraints. Since every configuration on $\overline{c_j c_{j+1}}$ is parameterized by t, this $\hat{T}$ can also be expressed as a function of t. Let this function be $f(t)$. Furthermore, when the robot $\mathbf{R}$ and $\mathbf{O}_i$ collide, they must reach a configuration c at the same time. Therefore, we also consider the relationship between t and T modeled by the function $g(t) : t = T$. See Fig. 2.

In Fig. 2, a bold (red) curve represents the earliest arrival time $f(t)$ and a black straight line represents $g(t)$. These two curves subdivide the space into interesting regions.

- For a point $p = (t, T > t)$, indicates situations that $\mathbf{O}_i$ reaches $c(t)$ later than t. No collisions will happen because when $\mathbf{O}_i$ reaches $c(t)$, the robot $\mathbf{R}$ already passes $c(t)$.
- The points $p = (t, T < f(t))$ indicates impossible situations that $\mathbf{O}_i$ needs to move faster than its maximum velocities in order to reach $c(t)$ at T.
- For a point $p = (t, f(t) < T < t)$ from the region above curve $f(t)$ but below curve $t = T$, $\mathbf{O}_i$ has the ability to

reach $c(t)$ earlier than **R**. In order to collide with **R**, $\mathbf{O}_i$ can slow down or wait at $c(t)$ until **R** arrives. We call this region the **collision region**.

Given that the robot **R** enters the path segment $\overline{c_jc_{j+1}}$ through one end point c_j at time t_j and leaves $\overline{c_jc_{j+1}}$ from the other endpoint c_{j+1} at time t_{j+1}, the earliest collision time ECT_{ij} is the t coordinate of left most point of the collision region between t_j and t_{j+1}. Therefore if this collision region is empty, **R** and $\mathbf{O}_i$ will not collide on $\overline{c_jc_{j+1}}$.

Based on what has been discussed so far, the most important step of estimating critical moment is to compute $f(t)$, the earliest moment when $\mathbf{O}_i$ reaches $c(t)$. The shape of function $f(t)$ depends on the type and the degrees of freedom of the robot and obstacles.

In the following sections, we will discuss three examples of how $f(t)$ can be formulated when: (1) both **R** and $\mathbf{O}_i$ are points, (2) **R** is a point and $\mathbf{O}_i$ is a polygon and (3) both **R** and $\mathbf{O}_i$ are polygons. From these examples, we can build up $f(t)$ for complex polygons even when rotation is considered.

IV. Point-Point Case

To warm up our discussion, we start with a point robot **R** and a point obstacle $\mathbf{O}_i$ without rotation. Let obstacle $\mathbf{O}_i$'s current pose p coincide with its reference point o and $c(t)$ is the pose of the robot at time t. The function $f(t)$ can be simply defined as

$$f(t) = |\overline{pc(t)}|/v_i \ . \qquad (1)$$

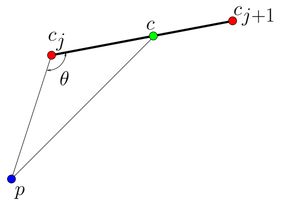


Fig. 3. When both $\mathbf{O}_i$ and **R** are points, their closest distance can be computed using Law of cosines in $\triangle pc_jc$.

Since **R** moves with a given velocity, $\overline{c_jc_{j+1}} \subset \Pi$ can be linearly interpolated and every point on $\overline{c_jc_{j+1}}$ is parameterized by $0 \leq \lambda \leq 1$. So the distance L between c_j and $c(t)$ is $L = |\overline{c_jc(t)}| = \lambda|\overline{c_jc_{j+1}}|$ and, the function f can be simply written as:

$$f(t) = \sqrt{L^2 + d^2 - 2dL\cos\theta}/v_i \qquad (2)$$

where $d = |\overline{pc_j}|$ and θ is the angle $\angle pc_jc_{j+1}$. This is illustrated in Fig. 3.

In order to compute the collision region, we need to find out the intersections of functions $f(t)$ and $g(t) : t = T$. By replacing $f(t)$ with t, we get a quadratic equation with only one variable $t = \sqrt{L^2 + d^2 - 2dL\cos\theta}/v_i$. By solving this equation, we can determine the collision region between $[t_j, t_{j+1}]$. If the collision region is empty, there will be no collision between $\mathbf{O}_i$ and **R** on path segment $\overline{c_jc_{j+1}}$. This operation is repeated on the other path segments until the ECT is detected; otherwise we report no collision between $\mathbf{O}_i$ and **R** on the entire path.

V. Point-Polygon Case

Now, we move on to the case where robot **R** is a point and obstacle $\mathbf{O}_i$ is a polygon that can translate and rotate around a given reference point o. Without loss of generality, let us focus on a moving segment $\overline{p_1p_2} \in \mathbf{O}_i$. As in point-point case, given a configuration $c(t) \in \overline{c_jc_{j+1}}$ which represents the location of **R** at time t, we are interested in solving $f(t)$, the earliest moment when $\overline{p_1p_2}$ hits $c(t)$.

To estimate the earliest collision time (ECT), we observe that $\mathbf{O}_i$'s rotation and translation can be considered separately. That is, $f(t)$ can be decomposed into translational and rotational components: t_T, the time that the point c needs to translate at velocity v_i, and t_R, the time that c needs to rotate at velocity ω_i. If we let the closest distance between $c(t)$ and $\overline{p_1p_2}$ be a function $\mathrm{d}(t)$ of time, we can compute ECT between $\overline{p_1p_2}$ and **R** moving from configuration c_1 to configuration c_2 using the following lemma.

Lemma 5.1: The ECT between $\overline{p_1p_2}$ and $\overline{c_1c_2}$ is:

$$\mathrm{ECT} = \arg\min_{t_R}\left(|t_R - t_T|\right) = \arg\min_{t_R}\left(|t_R - \mathrm{d}(t_R)/v_i|\right) \ , \qquad (3)$$

where $\mathrm{d}(t_R)$ is the distance between $c(t_R) \in \overline{c_1c_2}$ and segment $\overline{p_1p_2}$ when $\overline{p_1p_2}$ rotates $\theta = t_R\omega$ around o.

Proof: The key to this proof is the definition of the function $\mathrm{d}(t)$. In our analysis, $\mathrm{d}(t)$ depends on two cases: (1) $\overline{p_1p_2}$ and $c \in \overline{c_1c_2}$ are sufficiently far apart, and (2) $\overline{p_1p_2}$ and c are sufficiently close. Details of the analysis can be found in [31]. ■

In summary, to estimate the ECT of **R** and $\mathbf{O}_i$, we decompose $f(t)$ into translational and rotational components: t_T and t_R and solve the optimization problem in Lemma 5.1. Since both translation and rotation decrease the closest distance between **R** and $\mathbf{O}_i$, the time spend on translation t_T must equal the time spend on rotation t_R. Again, interested readers should refer to our technical report [31] for detail.

VI. Polygon-Polygon Case

In this section, we briefly discuss the case that both the robot **R** and the obstacle $\mathbf{O}_i$ are polygons. The robot **R** rotates around its center of mass and moves alone the designated path Π. Obstacle $\mathbf{O}_i$ undergoes unknown translation and rotation around a given reference point o.

Taking the same conservative advancement approach, we will focus our discussion on the motion strategy that an edge $\overline{q_1q_2}$ of $\mathbf{O}_i$ can take to collide with an edge $\overline{p_1p_2}$ of **R** at a given time t. Our main observation of computing their ECT is stated in the following lemma.

Lemma 6.1: Given two separating line segments $\overline{p_1p_2} \in \mathbf{R}$ and $\overline{q_1q_2} \in \mathbf{O}_i$, the earliest collision can only happen between an endpoint of $\overline{p_1p_2}$ and $\overline{q_1q_2}$ or an endpoint of $\overline{q_1q_2}$ and $\overline{p_1p_2}$. Collisions at the interior portion from both line segments can only happen after one of those two cases.
Proof: See detailed proof in technical report [31]. ■

Essentially, Lemma 6.1 allows us to determine the ECT of two line segments from only two instances of *point-polygon case* discussed in Section V. Given that **R** and $\mathbf{O}_i$ are composed of n and m line segments, respectively, their ECT can be determined via $2mn$ point-polygon case analysis.

VII. Planning Motion Using Predicted Collision

So far we assume that the robot only stays on a given path. In this section, we show how to use the predicted collision in a motion planner. It is important to note that this RRT-based planner discussed below is merely an example to show how earliest collision time (ECT) can be used. Other planners, such as PRM-based planners, can also be combined with ECT.

In general, there are two desirable properties when a robot plans a path. First, a path should bring the robot near the goal. Second, the path should remain safe (valid) for as long as possible. With these two properties in mind, we propose to augment RRT with predicted collision. More specifically, the RRT is constructed as usual but each path from the root to a leaf is now associated with an ECT. The best path is then a path in the RRT that has the *latest* ECT while still reduces the geodesic distance between the robot and the goal.

VIII. Experimental Results

We implemented the collision prediction method in C++ using Eigen linear algebra library and NLopt library. Experimental results reported in this paper are obtained from a workstation with two Intel Xeon E5-2630 2.30GHz CPUs and 32GB memory. We tested our implementation in four environments shown in Figs. 1 and 4. These environments contain both static and dynamic obstacles. For a dynamic obstacle, its motion is simulated using Box2D physics engine by exerting random forces. The robot knows the locations of static obstacles and the maximum translational velocity and angular velocity of dynamic obstacle. The only way that the robot knows the pose of a dynamic obstacle is through its (simulated) onboard sensors. The best way to visualize the environments is via animation. We encourage the reader to view the videos at *http://masc.cs.gmu.edu/wiki/ECT*.

A. Compare to a Fixed-Time Strategy

In our first experiment, we compare two planning strategies: One replans adaptively based on collision prediction using augmented RRT (see Section VII), and the other replans periodically at fixed time interval using regular RRT.

Fig. 5 shows the *success rate* and *number of replans* obtained from environments in Fig. 4. The *success rate* is the number of runs that robot reaches the goal over the total number of runs, and the *number of replans* is the number of times that the robot replans to reach the goal. The maximum translational velocity of an obstacle is set to $2m/s$ and the maximum angular velocity is set to 3 $radians/s$. The experiments are conducted for multiple situations when robot's velocity is 1, 2, 4, 8 and $16m/s$. Each data point is collected over 500 runs (i.e. 100 runs for each environment).

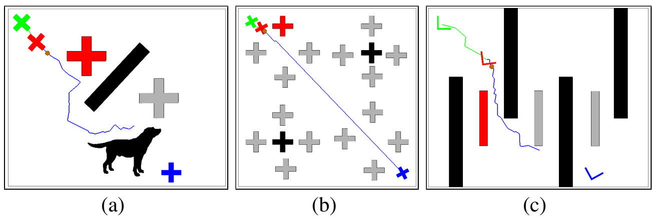


Fig. 4. (a-c) Polygon-polygon envi-ronments. In all environments, the green robot and blue robot indicate start configuration a nd g oal c onfiguration, re spectively. Th e re d ro bot indicates the current configuration a nd t he o bstacles w hich c ause e arliest collisions are colored in red. Black obstacles are static and light grey obstacles are dynamic.

Success Rate and Number of Replans. From the plots in Fig. 6, we show that our approach using predicated collision helps the robot achieve nearly optimal success rate with a small number of replans. First, let us look at Fig. 6 (left). We see that the success rate of the proposed method is almost identical to or even better than the fixed-time strategy with very high (and almost unrealistic) replanning frequency (i.e. replan every 0.05 sec.). This is especially clear when the robot's velocity is greater than $2m/s$. However, frequent updates introduce a large number of replans. As shown in Fig. 6 (right), in order to provide a success rate similar to the proposed method, the fixed-time strategy needs to replan around 100 times more.

IX. Conclusion

In this paper, we proposed an adaptive method that predicts collisions for obstacles with unknown trajectories. We believe that this collision prediction has many potential usages and advantages. Similar to collisions detection in the setting of known obstacle motion, we have shown that collision prediction allows the robot to evaluate the safety of each edge on the extracted path with unknown obstacle motion. When the robot travels on a predetermined path, collision prediction enables adaptive repairing period that allows more robust and efficient r eplanning. C omparing t o a planning strategy that replans periodically at *fixed t ime i nterval*, our experimental results show strong evidences that the proposed method significantly reduces the number of replans while maintaining higher success rate of finding a valid path.

References

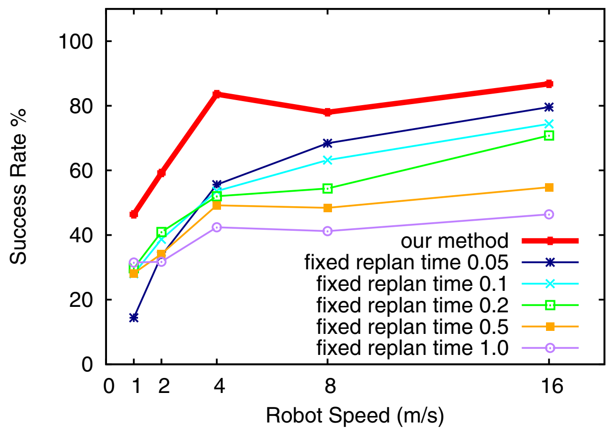
[1] L. Jaillet and T. Simeon, "A prm-based motion planner for dynamically changing environments," in *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, 2004, pp. 1606–1611.

[2] M. Kallman and M. Mataric, "Motion planning using dynamic roadmaps," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 5. IEEE, 2004, pp. 4399–4404.

[3] T.-Y. Li and Y.-C. Shie, "An incremental learning approach to motion planning with roadmap management," in *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2002, pp. 3411–3416.

[4] D. Ferguson, N. Kalra, and A. Stentz, "Replanning with rrts," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. IEEE, 2006, pp. 1243–1248.

[5] O. Khatib, "Real–time obstacle avoidance for manipulators and mobile robots," *Int. Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.



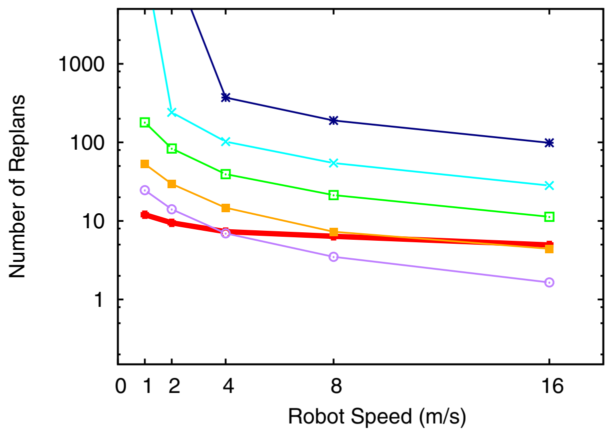


Fig. 5. Compare our method to the fixed-time strategy obtained from environments in Fig. 1(b) and in Fig. 4. Each data point is averaged over 500 runs. In the fixed-time s trategy, t he r obot replans every 0.05, 0.1, 0.2, 0.5 and 1.0 seconds. Notice that the *y*-axis is in logarithmic scale.

[6] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime dynamic a*: An anytime, replanning algorithm," 2005.

[7] J. van den Berg, D. Ferguson, and J. Kuffner, "Anytime path planning and replanning in dynamic environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2006, pp. 2366 – 2371.

[8] J. van den Berg and M. Overmars, "Planning the shortest safe path amidst unpredictably moving obstacles," in *Proc. Int. Workshop Alg. Found. Robot.(WAFR)*, 2006.

[9] M. Wzorek, J. Kvarnstrom, and P. Doherty, "Choosing path replanning strategies for unmanned aircraft systemsun," 2010.

[10] V. Hayward, S. Aubry, A. Foisy, and Y. Ghallab, "Efficient collision prediction among many moving objects," *Internat. J. Robot. Res.*, vol. 14, no. 2, pp. 129–143, 1995.

[11] P. M. Hubbard, "Collision detection for interactive graphics applications," Ph.D. dissertation, 1995.

[12] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment: A collision cone approach," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 28, no. 5, pp. 562–574, 1998.

[13] H. K. Kim, L. J. Guibas, and S. Y. Shin, "Efficient collision detection among moving spheres with unknown trajectories," *Algorithmica*, pp. 195–210, 2005.

[14] 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 *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 3931–3936.

[15] P. Henry, C. Vollmer, B. Ferris, and D. Fox, "Learning to navigate through crowded environments," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 981–986.

[16] A. Wu and J. P. How, "Guaranteed infinite horizon avoidance of unpredictable, dynamically constrained obstacles," *Autonomous Robots*, pp. 227–242, 2012.

[17] N. Du Toit and J. Burdick, "Robotic motion planning in dynamic, cluttered, uncertain environments," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 966–973.

[18] K. Hauser, "Randomized belief-space replanning in partially-observable continuous spaces," *Algorithmic Foundations of Robotics IX*, pp. 193–209, 2011.

[19] Z. Shiller, O. Gal, and A. Raz, "Adaptive time horizon for on-line avoidance in dynamic environments," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 3539–3544.

[20] B. D. Luders, G. S. Aoude, J. M. Joseph, N. Roy, and J. P. How, "Probabilistically safe avoidance of dynamic obstacles with uncertain motion patterns," MIT Aerospace Control Laboratory: Technical Reports, Tech. Rep., 2011.

[21] T. Fraichard and H. Asama, "Inevitable collision states. a step towards safer robots?" in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 1. IEEE, 2003, pp. 388–393.

[22] L. Martinez-Gomez and T. Fraichard, "Collision avoidance in dynamic environments: an ics-based solution and its comparative evaluation," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 100–105.

[23] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J. How, and G. Fiore, "Real-time motion planning with applications to autonomous urban driving," *Control Systems Technology, IEEE Transactions on*, vol. 17, no. 5, pp. 1105–1118, 2009.

[24] S. Bouraine, T. Fraichard, and H. Salhi, "Relaxing the inevitable collision state concept to address provably safe mobile robot navigation with limited field-of-views in unknown dynamic environments," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 2985–2991.

[25] E. Lalish and K. Morgansen, "Decentralized reactive collision avoidance for multivehicle systems," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*. IEEE, 2008, pp. 1218–1224.

[26] A. Bautin, L. Martinez-Gomez, and T. Fraichard, "Inevitable collision states: A probabilistic perspective," in *Proc. of IEEE Int. Conf. on Robotics and Automation*, Anchorage, AK, 2010, pp. 4022 – 4027.

[27] J. Cohen, M. C. Lin, D. Manocha, and M. Ponamgi, "I-collide: An interactive and exact collision detection system for large-scale environment," in *Symposium on Interactive 3D Graphics*, 1995, pp. 189–196.

[28] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBTree: A hierarchical structure for rapid interference detection," *Computer Graphics*, vol. 30, pp. 171–180, 1996.

[29] D. Baraff, "Curved surfaces and coherence for non-penetrating rigid body simulation," *Comput. Graph.*, vol. 24, no. 4, pp. 19–28, 1990.

[30] J. K. Hahn, "Realistic animation of rigid bodies," *Comput. Graph.*, vol. 22, no. 4, pp. 299–308, 1988.

[31] Y. Lu, Z. Xi, and J.-M. Lien, "Conservative collision prediction among polygons with unknown motion," George Mason University, Tech. Rep. GMU-CS-TR-2013-4, 2013.