A Sampling-Based Approach to Probabilistic Pursuit Evasion

Aditya Mahadevan and Nancy M. Amato

***Abstract*— Probabilistic roadmaps (PRMs) are a sampling-based approach to motion-planning that encodes feasible paths through the environment using a graph created from a subset of valid positions. Prior research has shown that PRMs can be augmented with useful information to model interesting scenarios related to multi-agent interaction and coordination.**

Pursuit evasion is the problem of planning the motions of one or more agents to effectively track and/or capture an initially unseen evader in an environment.

Unlike prior probabilistic approaches that assume the environment is partitioned into convex cells or square grids, we present a sampling-based technique that allows us to generalize the problem to an arbitrary partitioning of the environment. We then show how PRMs can exploit this method using Voronoi diagrams. We discuss the theoretical underpinnings of this approach and demonstrate its validity experimentally.

I. INTRODUCTION

Pursuit evasion is a growing field in multi-agent robotics, with applications as varied as search and rescue, home care, mobile sensor networks and military combat. These applications have inspired an equally diverse set of approaches using techniques such as game theory, graph theory [1] and computational geometry [2]. Although the roots of this problem are in graph theory, there is an increasing focus on modeling realistic scenarios, and on tighter coordination between abstract planning and the uncertainty arising from sensors and effectors.

We have previously explored how robots can augment probabilistic roadmaps with information about other agents and their environment to achieve realistic behaviors under various conditions [3], [4], [5]. Here, we focus on the scenario of maintaining information about other agents out of view.

A. Overview

Probabilistic roadmaps [6], [7], [8], [9] are a sampling-based approach to motion planning. They are graphs whose vertices are sampled from the free space and therefore represent 'legal' robot configurations. Edges connect nearby vertices between which there is a collision-free path. A roadmap can be queried multiple times to find paths between two arbitrary configurations. In 2D and 3D environments, valid configurations may represent robot positions and orientations.

Pursuit evasion is the problem of planning the motion of one or more agents in an environment containing obstacles to effectively find and capture an unknown, possibly adversarial evader. The environment is often discretized into a set of connected regions (partitions). In these cases, the pursuit evasion problem becomes that of visiting partitions in an order that either guarantees that the evader will be captured or minimizes the expected time to capture.

We focus on a probabilistic formulation of the pursuit evasion problem. Under this formulation, probability values are assigned to each partition and to the edges connecting partitions (reflecting uncertainty on the part of pursuers about the evader's current position and actions, respectively). The goal of the pursuers is to reduce the uncertainty about the evader's position, ultimately leading to the latter's discovery and capture. Most approaches [10], [11], [12], [13] make closely related assumptions about how the environment is partitioned and the sensing and movement capabilities of all agents in order to assign transition probabilities.

In this paper, we attempt to generalize the problem to include arbitrary discretizations of the environment, with minimal assumptions on the evader's movements. We then demonstrate how probabilistic roadmaps can be used both to discretize the environment and to plan the agents' motions.

B. Contributions

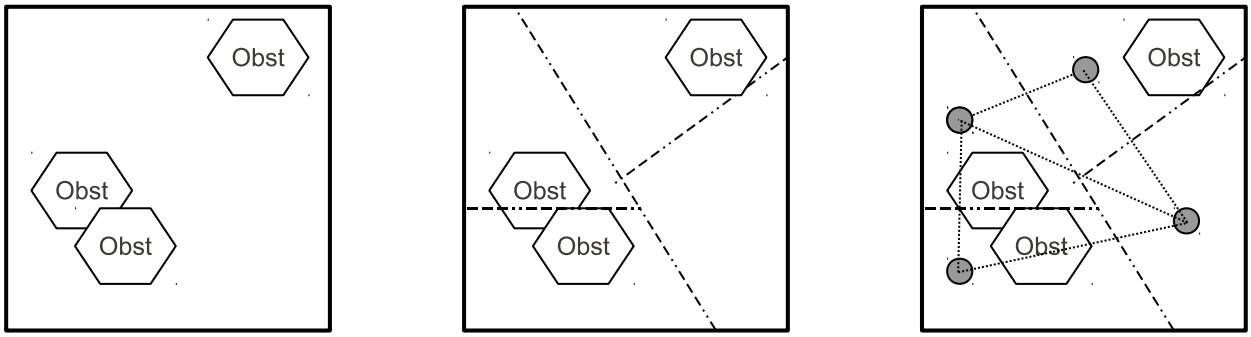
We make the following contributions to the probabilistic formulation of the pursuit evasion problem:

- A sampling-based algorithm for online computation of transition probabilities for an environment that is discretized into partitions of arbitrary size and shape. We also show tight bounds when the discretization is a Voronoi tessellation.
- Extension of the probabilistic pursuit evasion framework to arbitrary discretizations of the environment. This is a direct result of our sampling-based approach. Prior work discretizes the environment into regular grid cells or convex cells using the pursuers' visibility.

As touched on above, and discussed in more detail in the paper, we believe the decoupling of the partitioning from the sensing and movement constraints of the agents provides advantages over other probabilistic approaches. The results presented illustrate the effectiveness of the strategy and also its sensitivity to a number of factors such as the size and type of the roadmap and the update frequency.

This research supported in part by NSF awards CRI-0551685, CCF-0833199, CCF-0830753, IIS-096053, IIS-0917266 by THECB NHARP award 000512-0097-2009, by Chevron, IBM, Intel, Oracle/Sun and by Award KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST).

A. Mahadevan and N.M. Amato are with the Parasol Laboratory, Department of Computer Science and Engineering, Texas A&M University, College Station, TX, 77843-3112, USA. {aditya.mahadevan,amato} @tamu.edu



(a) An environment in 2D space

(b) Arbitrary partitioning of the environment

(c) Graph with partitions represented as nodes and adjacencies between partitions as edges

Fig. 1: An arbitrary discretization of an environment, and the resulting graph.

II. RELATED WORK

Pursuit evasion has evolved considerably from its original formulation as a graph-theoretic problem and spawned a number of variants and approaches. We review relevant literature in the field, with a particular focus on probabilistic approaches, of which ours is one.

A. Basic Pursuit Evasion Problems

The earliest work on pursuit evasion in graphs was done by T.D. Parsons [1], who defined the search number of a graph as the minimum number of pursuers necessary to guarantee capture of an evader moving arbitrarily fast through the edges. In graph-based approaches, visibility is restricted to the pursuers' current or adjacent vertices/edges [14].

Some of the first descriptions of the problem in polygonal environments were presented by Lavalle and Guibas, such as in [2], [15]. In these papers, the pursuer was assumed to have infinite visibility along a straight line, limited only by obstacles and the environment boundary. The evader was once again adversarial, omniscient and arbitrarily fast.

Since these seminal papers, a number of variants to the problem have been proposed that place constraints on the sensing abilities of the pursuers and evader, by allowing communication and coordination, and by varying the nature of the environment. The problem of restricting or limiting the amount of visibility is considered in [16], with the field of view being variable in [17]. The problem is extended to teams of robots with different kinds of vehicles and sensing information in [18]. Height maps, which also restrict agent visibility, are described in [19].

B. Roadmap-Based Approaches

Prior roadmap-based approaches include our most recent prior work [3], [4], [5], all of which extend the pursuit evasion problem to 3D structures and complex environments such as terrains and the presence of crowds. We focused exclusively on 3D structures in [4] and first introduced the idea of augmenting the roadmap with probability information. However, we did not explain how to calculate transition probabilities, an issue that we address in this paper.

C. Probabilistic approaches

Probabilistic formulations of the pursuit evasion problem [10], [11], [12], [13] are especially related to the work presented in this paper. These approaches work in a way similar to ours: they discretize the environment in some way into cells; define a probability distribution of the evader over them; assign values to each edge of the resulting graph representing the probability that the evader transitions between respective cells; and at every timestep multiply the *a priori* probability at each cell by the transition probabilities to obtain the posterior probability distribution of the evader at the next timestep. However, the approach we present here has the advantage of working with arbitrary discretizations of the environment, evader models and sensing abilities. In contrast, these prior approaches assume a particular discretization, evader model and/or sensor constraints.

A probabilistic framework to pursuit evasion is presented in [10]. It considers an online problem in which map-building and pursuit are combined. Two greedy policies governing the pursuers' movements are compared – a local-max and a global-max policy. The environment is discretized into regular grid cells and places constraints on the positions of pursuers and evaders – they cannot occupy the same cell as an obstacle, and no two pursuers can occupy the same cell. This implies that the resolution of the cell grid must be fine enough to account for obstacle and agent geometries, that some free space is off-limits to agents because it shares the same grid as an obstacle of unknown size, or that each obstacle occupies exactly one grid cell. Capture occurs when a pursuer lands in the same cell as an evader. All agent actions are assumed to take an equal amount of time, represented as a timestep. Cells adjacent to a given cell are assumed to be reachable from that cell in a single timestep. The evader is assumed to follow a Markov model. The transition probability from one cell to an adjacent unoccupied cell is simply the reciprocal of the number of adjacent unoccupied cells; this arises naturally from the Markov model assumption and the fact that all grid cells have equal size. Furthermore, the conditional probability of

an evader remaining in a particular cell across consecutive timesteps is assumed not to decay by more than a certain fraction, an implied limit on the evader's maximum velocity.

In [11], pursuit evasion is considered as a nonzero-sum game quantized in space and time, in which pursuers and the evader try to maximize and minimize the likelihood of capture respectively. The evader is adversarial, and has access to all information available to the pursuers. Both players follow stochastic policies. As in [10], the game takes place on a 2D grid with square cells, and agents cannot occupy cells that contain obstacles. Capture occurs when a pursuer occupies the same cell as an evader. The transition probability for an agent at a particular cell is equivalent to the probability that an agent chooses a particular action when at that cell. There is again the notion of one-step reachable cells, i.e., the set of cells adjacent to the one currently occupied by the agent. Any action is a transition into one of these cells.

An interesting POMDP approach is presented in [13] that unifies target searching and target following. In particular, a sampling-based algorithm is presented that attempts to approximate the reachable space under optimal policies. The particular scenario is that of an assistive robot tracking an elderly person, attempting to minimize the time taken to reach the person if summoned while also conserving energy. A grid-based discretization is employed in order to make the POMDP formulation feasible. The target's position is a probability distribution over grid cells, and its movement is a set of transition probabilities between grid cells. An agent always lies in some state and aims to take a sequence of actions that maximizes its total expected reward. Due to uncertainty in actions, there is a transition probability between states rather than grid cells, denoting the likelihood that the agent at state s reaches a state s' under an action a.

III. A SAMPLING-BASED APPROACH TO TRACKING

In this section, we present our approach to determining transition probabilities between partitions of an environment given a model of how the evader could act under its current situation at any timestep.

A. Arbitrary Partitions of An Environment with Obstacles

Unlike prior approaches to probabilistic pursuit-evasion that require the environment to be partitioned according to strict constraints, the method we present in this paper allows us to partition the environment in any manner. We define partitions as follows.

Let $\mathcal{W}$ represent a bounded 2D space on which agents and obstacles can exist. An example is given in Fig. 1(a). Suppose there is a collection of subsets $W = \{w_1, w_2, ..., w_n\} \in \mathcal{W}$ such that $\bigcup_{i \in n} w_i = \mathcal{W}$ and $\bigcap_{i \in n} w_i = \emptyset$. Now the set W is an arbitrary partitioning of $\mathcal{W}$ into n subsets, such that

- Each subset is a bounded, simply connected space (such as a convex polygon),
- No subset $w \in W$ overlaps with any other, and
- Every point in $\mathcal{W}$ is contained in a single subset.

An example is illustrated in Fig. 1 (b). The last two properties ensure that the subsets cover the entire space but do not overlap. We henceforth refer to each subset as a partition.

We also require that within every partition, the *free space* (i.e. the space not occupied by an obstacle) should be simply connected[1]. This will allow us to represent each partition as a node in a graph.

B. Transitions Between Partitions

We now show how pursuers in $\mathcal{W}$ can make use of a representation of the partitions of $\mathcal{W}$ to track evaders.

We create a graph $G_W = (V, E)$ with $|V| = n$ such that each vertex $v \in V$ maps uniquely to one partition in W and there is an edge $e = v_i, v_j \in E$ between two vertices v_i, v_j if and only if the corresponding partitions in W share a boundary in $\mathcal{W}$, at least partially lying in free space, as shown in Fig. 1(c). Hence an edge between two nodes represents the existence of a direct collision-free path between corresponding partitions.

Since the partitions cover the entire space with no overlap, any agent will exist in exactly one partition at any instant in time. We define a probability distribution over the set of partitions denoting the likelihood that an agent is at a particular one at timestep t, given by $p(w_i, t)$ for $w_i \in W$ such that $\sum_{w_i \in W} p(w_i, t) = 1$.

We can model the likelihood of an agent transitioning from one partition to another at any timestep by assigning transition probabilities to the edge $(v_i, v_j) \in E$ corresponding to every pair $v_i, v_j \in V$, where v_i and v_j correspond to partitions w_i and w_j, respectively. The transition probability $T_t(w_i, w_j)$ is the conditional probability that an agent will move from w_i to w_j in timestep $t + 1$ given that the agent is in w_i at timestep t, i.e., $T_t(w_i, w_j) = p(w_j, t + 1 | w_i, t)$.

The accuracy of our tracking depends entirely on how well we assign these transition probabilities.

C. Determining Transition Probabilities

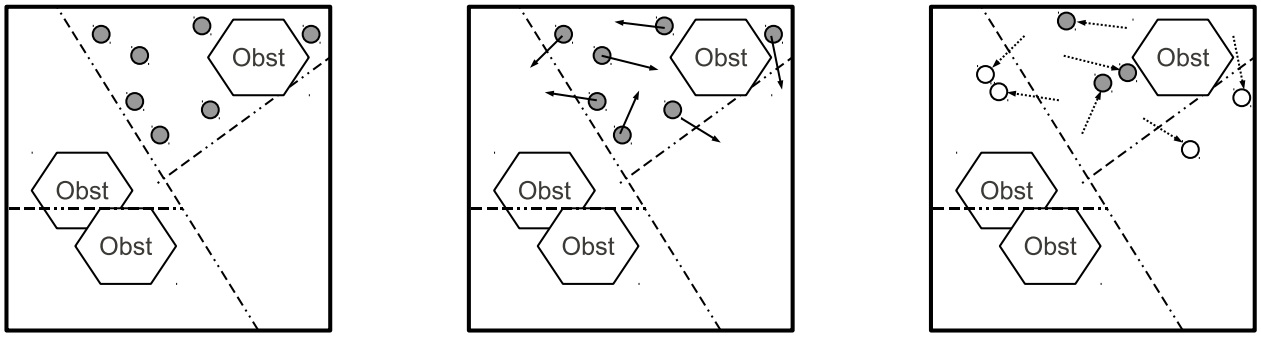
We now present an approach to assigning these transition probabilities by sampling from an agent's possible positions and a known model describing the agent's actions.

1) Evader Movement Model: We make the following assumptions about the target:

- In a single timestep, it travels along a straight line
- It has a known or observable bounded velocity vel_{max}
- It can cross at most one boundary between partitions per timestep[2]

[1] In practice, this requirement is easy to satisfy simply by dividing the environment into a sufficiently large number of partitions.

[2] This assumption does not conflict with our claim to handle arbitrarily partitioned environments as long as we recognize and account for the relationship between partition size/shape, timestep size and velocity. Consider a straight line intersecting a partition exactly twice, such that the line segment incident on the intersection points spans the 'narrowest' part of the partition. An agent traveling along the line from one intersection point to the other will completely traverse the line segment in one timestep if the distance it can travel in one timestep (i.e., its velocity, expressed as a function of the timstep size) is at least equal to the length of the line segment. We can therefore handle the above assumption by simply setting a timestep size that is small enough.



(a) Generate samples (shown as shaded circles) in each partition

(b) Push samples by a vector obtained from sampling on the probabilistic evader model $\mathcal{M}$

(c) Some fraction of samples (shaded) stay in same partition, while others (unshaded) cross to adjacent partitions

Fig. 2: Illustration of the sampling-based approach. (a) Random samples are generated in each partition. (b) Each sample is pushed in a random direction according to the probability distribution representing the evader model $\mathcal{M}$ (c) The fraction of samples that remain in the same partition are an estimate of the conditional probability that the evader remains in that partition in the next timestep. Similarly, the fraction that cross a particular boundary to an adjacent partition estimate the conditional probability of crossing into that partition from the original one.

The space of possible displacements available to a target is given by $\mathbb{R} \times \mathbb{S}$ where $\mathbb{S} = [-\pi, \pi]$, and a displacement can be represented by a 2D vector $\vec{d}$ with magnitude d and direction θ where $d \in \mathbb{R}, 0 \leq d \leq vel_{max}$, and $\theta \in \mathbb{S}$.

We describe a random vector in this space using the continuous random variables D and Θ. We define a model of movement over this space as a joint probability distribution function, i.e., a mapping $\mathcal{M} : \mathbb{R} \times \mathbb{S} \mapsto [0, 1]$ such that $\mathcal{M}_{D,\Theta}(d, \theta) = p(D \leq d, \Theta \leq \theta)$. For instance, under a random walk model in open space, D and Θ are independent and follow uniform distributions on $[0, vel_{max}]$ and $[-\pi, \pi]$ respectively. Our method will also work with more sophisticated models as long as they are expressible as a probability distribution. For example, $\mathcal{M}$ could even be a conditional probability distribution given the state of the pursuers and/or evader.

2) Sampling to Determine Transition Probabilities: We can now use this model to randomly sample from the agent's possible transitions between partitions in a single timestep t, as follows.

- First, we randomly sample the agent's possible current positions in each partition w_i.
- Next we push each sample s by a vector $\vec{d}$ sampled from the probability distribution $\mathcal{M}$.
- We then determine the partition w_j containing the resulting resulting sample s'. Since we assume an agent can cross at most one partition per timestep, w_j will be adjacent to w_i.
- We assign a transition probability to the edge (v_i, v_j) in G_W that is equal to the fraction of samples in w_i that have been pushed to w_j.

This process is outlined in Algorithm 1, the generic ComputeTransitionProbabilities algorithm (CTP-Generic). The fraction of samples in a partition w_i that remain in it after being pushed by $\vec{d}$ is an estimate of $p(w_i, t + 1|w_i, t)$, i.e., the conditional probability that an agent in a given partition remains in it in the next timestep. Likewise, the fraction of samples that cross over into an adjacent partition w_j is an estimate of $p(w_j, t + 1|w_i, t)$, i.e., the conditional probability that an agent in the first partition crosses over into the adjacent partition in the next timestep. Each conditional probability is stored on the directed edge connecting the nodes that represent the two partitions.

Given sufficient samples and a reasonably accurate movement model, we can have good estimates of transition probabilities for any pair of partitions. Significantly, this method of computing transition probabilities can be applied to partitions of any size and shape, as long as they satisfy our definition of a partition. This includes partitions used in [10], [11], [12], [13].

It should be noted that we discard any sample s that is in collision with obstacles (since it is not a valid position of the target). We also discard any sample in the visibility region of the pursuer(s) since an evader at such a position would have been seen. (To account for uncertainty in the sensor model, we could discard such samples with a certain probability.)

D. Tracking Process

We now describe how we can use these estimates to infer a target's position over time, given an initial position at timestep $t = 0$. Whenever a target is visible, we assign an *a priori* probability of 1.0 to the node representing the target's current partition and 0.0 to all others. Otherwise, we multiply the *a priori* probability at each node by the conditional probabilities stored on the outgoing edges. The resulting probability is added to the destination node before the next timestep. In this manner, every node "propagates" its probability value to itself and to adjacent nodes. The effect

Algorithm 1 CTP-GENERIC

Input. A set P of randomly sampled sites (roadmap nodes) in the environment $\mathcal{W} = \mathcal{W}_{free} \cup \mathcal{W}_{obst}$.
A set W of arbitrary partitions of $\mathcal{W}$, and a graph $G_W = (V, E)$ describing their adjacencies.
A probability distribution $\mathcal{M}$ representing the evader model.
Output. Transition probabilities for every pair of adjacent partitions.

for $w_i \in W$ **do**
  $v_i \leftarrow vertex\ in\ G_W\ representing\ partition\ w_i$
  for $w_j \in w_i \cup adj(w_i)$ **do**
    $v_j \leftarrow vertex\ in\ G_W\ representing\ partition\ w_j$
    $e(v_i, v_j).weight \leftarrow 0$
  end for
  $S \leftarrow$ SAMPLEPOINTS $in\ \mathcal{V}(w_i) \cap \mathcal{W}_{free}$
  for $s \in S$ **do**
    $\vec{v} \leftarrow$ SAMPLEVECTOR $according\ to\ \mathcal{M}$
    $s' \leftarrow$ PUSH $s\ by\ \vec{v}$
    $w_j \leftarrow partition\ containing\ s'$
    $v_j \leftarrow vertex\ in\ G_W\ representing\ partition\ w_j$
    $e(v_i, v_j).weight \leftarrow e(p_i, p_j).weight + \frac{1}{|S|}$
  end for
end for

is to disperse the probability across an increasing number of nodes every timestep. The probability value at a particular node at any timestep is the likelihood of finding the target in the corresponding partition at that timestep.

IV. ROADMAP-BASED TRACKING

In the previous subsections, we proposed a general approach to tracking in arbitrarily discretized environments using a sampling-based method to calculate transition probabilities. We now explain how this approach can be combined with probabilistic roadmaps to enable tracking of evaders in complex environments.

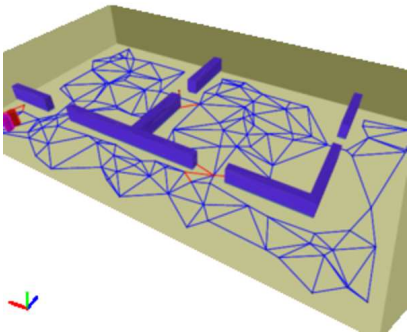
A. Spatial Properties of Probabilistic Roadmaps

In 2D (or 3D), probabilistic roadmaps are spatial graphs embedded in the environment that they represent. Hence each node has a unique set of coordinates, and the edges between nodes have lengths corresponding to the physical distance between the endpoints.

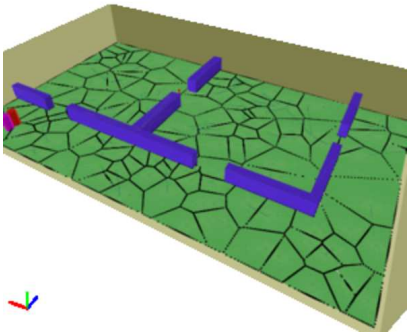
We can exploit this spatial embedding to produce an elegant partitioning of the environment as follows.

B. Roadmap-Based Partitioning

Given a distribution of points (sites) on a plane, the Voronoi region of each site is the set of all points on the plane nearer to that site than to any other. The Voronoi diagram of the set of sites is the partitioning of the plane induced by assigning each point on the plane to the site nearest to it. There is exactly one site in each region and one region per site. All we require to perform such a partitioning is the ability to efficiently and deterministically calculate the relative distances from a given point to all sites.



(a) Probabilistic roadmap for a 2D environment



(b) Voronoi tessellation of the environment

Fig. 3: Voronoi tessellation of the environment using roadmap nodes. (a) Nodes are sampled randomly in the environment. The sampling approach is used to determine pairs of nodes whose Voronoi regions are adjacent. These node pairs are connected by an edge. (b) The Voronoi tessellation is completed.

We describe this idea formally using the following notation. Let $P := p_1, p_2, ...p_n$ be a set of n distinct points in a plane, which are the sites. The Voronoi diagram of P is the subdivision of the plane into n partitions, one for each site in P, such that a point q lies in the partition corresponding to a site p_i iff $dist(q, p_i) < dist(q, p_j)$ for each $p_j \in P$ with $j \neq i$.

We denote the Voronoi diagram of P by $Vor(P)$, and the particular partition corresponding to a site p_i by $\mathcal{V}(p_i)$.

The nodes of a probabilistic roadmap on a 2D plane form a collection of sites from which we can construct a Voronoi diagram. Hence, we can use the nodes themselves to partition the environment completely such that each partition contains a roadmap node, and each node represents a single partition. We then add an edge between every pair of nodes whose partitions are adjacent, i.e., they share a boundary.

C. Computing Transition Probabilites for Roadmaps

We can now apply the same method we described in the generic case to the Voronoi partitions. The cornerstone of the approach we described is the ability to calculate the

membership of samples, i.e., to determine which partition a sample belongs to. This is known as the point-location problem.

1) Point Location Using Nearest Neighbor Query: Since partitions can be of arbitrary shape and size, this is a non-trivial task in general. However, when partitions are Voronoi regions, we have a convenient way to determine which partition a sample s belongs to. It is simply the partition represented by the roadmap node closest to s, i.e., the node that is the *nearest neighbor* of the query point s. This follows from the relationship between Voronoi diagrams and nearest neighbor queries. This is beneficial because we can exploit existing techniques for efficient nearest-neighbor queries, and we need not construct explicit representations of the partitions – which is particularly beneficial in the presence of obstacles in the environment.[3]

2) CTP-PRM: We outline this process in the Compute-TransitionProbabilities Algorithm (Algorithm 2) for probabilistic roadmaps (CTP-PRM). The steps are essentially the same as in CTP-Generic, but for CTP-PRM we have tight bounds on the running time.

Processing each partition involves 3 main parts: resetting the weights (steps 2-4), sampling points (step 5) and pushing samples to get transition probabilities. These are repeated for each partition. Steps 7, 8 and 10 take $O(1)$ time. Step 9 is linear if we use the naive approach of computing distances to every site and taking the smallest-distance site. However, with the use of Voronoi diagrams, the query time has an upper bound that is logarithmic on the number of sites. Querying a point location data structure has a known lower bound of $O(logn)$ for n points, so this step has a tight bound of $\Theta(log|P|)$. The total complexity for the for-loop at step 6 is therefore $\Theta(|S|log|P|)$. Step 5 is $\Theta(|S|)$. Steps 2-4 trivially have a worst-case upper bound of $(|P|^2)$; however, since the Voronoi partitioning results in a planar graph, this has an amortized tight bound of $\Theta(|P|)$ (the number of edges of a planar graph is linear in the worst case).

Since we perform these steps for each partition, the complexity of the algorithm overall is $\Theta(|S||P|log|P|)$. Step 9 is the dominant step.

Algorithm 2 CTP-PRM

Input. A set P of randomly sampled sites (roadmap nodes) in an environment represented by $\mathcal{W} = \mathcal{W}_{free} \cup \mathcal{W}_{obst}$. A probability distribution $\mathcal{M}$ representing the evader model.
Output. Transition probabilities for every pair of adjacent partitions.

1: **for** $p_i \in P$ **do**
2: **for** $p_j \in p_i \cup adj(p_i)$ **do**
3: $e(p_i, p_j).weight \leftarrow 0$
4: **end for**
5: $S \leftarrow$ SAMPLEPOINTS $in\ \mathcal{V}(p_i) \cap \mathcal{W}_{free}$
6: **for** $s \in S$ **do**
7: $\vec{v} \leftarrow$ SAMPLEVECTOR $according\ to\ \mathcal{M}$
8: $s' \leftarrow$ PUSH s *by* $\vec{v}$
9: $p_j \leftarrow$ CLOSESTSITE (s')
10: $e(p_i, p_j).weight \leftarrow e(p_i, p_j).weight + \frac{1}{|S|}$
11: **end for**
12: **end for**

V. EXPERIMENTS

We validate our proposed method with a series of experiments meant to highlight both the effectiveness and usefulness of the method. In particular, we explore the effect of sample size, update frequency and type of roadmap on the performance of the method.

A. Experimental Setup

We test our algorithm on two environments – an office-like environment with straight, thin, orthogonal walls and a rocky environment with obstacles of uneven size and placement.

[3]When obstacles are present, we can modify the distance function used in the nearest-neighbor query. If the straight line between a node and the query point intersects an obstacle, the distance between these two points is taken to be infinite. Otherwise, it is the Euclidean distance. The effect of this definition of distance is that a query will return the closest *visible* node.

For all experiments, the agents are initially placed such that both the pursuer and evader are visible to one another. The evader is faster than the pursuer, so that the former is initially able to escape, and the pursuer immediately relies on the probability model. Since they have the same view radius, once out of sight the evader does not know where the pursuer is until they both see each other again. The pursuer assumes that the evader follows a Markov model of movement. If the evader is visible, the pursuer moves directly towards it. Otherwise, it follows a policy of moving to the partition with the highest probability of containing the evader, whenever it needs a new goal position. The goal position is updated when the current goal position is reached.

For all experiments, we take the average of 5 runs using different random seed values.

B. Effect of Sample Size

Since the proposed method is sampling-based, we expect that the accuracy of the method will increase as the number of samples increases. We test this hypothesis by varying the sample size while keeping the update period (time between updates, measured in timesteps) constant.

We use the percentage of time the evader is hidden as a measure of performance of the probability model, since a less accurate model is likely to result in the evader staying hidden for longer periods of time once it escapes.

Our results, plotted in Fig. 4, show that the accuracy does initially increase with the sample size, as indicated by the decreasing percentage of time the evader is hidden, but eventually levels off. This is exactly what we would expect. Moreover, the effectiveness of the method is clearly seen in the sizeable difference between its worst performance (with a sample size of 1), when the evader is hidden roughly 93% of the time, to its best performance, when the evader is only hidden roughly 75% of the time.

Time hidden (%)				
	Env.1 (62 nodes)		Env.2 (114 nodes)	
Samples (time between updates)	Roadmap 1	Roadmap 2	Roadmap 1	Roadmap 2
5 (10)	.870(.031)	.909(.029)	.925(.022)	.909(.038)
10(20)	.866(.034)	.881(.020)	.932(.020)	.924(.024)
15(30)	.886(.032)	.912(.032)	.940(.026)	.912(.028)
20(40)	.882(.012)	.856(.063)	.931(.029)	.913(.010)
25(50)	.893(.056)	.880(.026)	.931(.021)	.920(.032)
30(60)	.870(.020)	.915(.026)	.925(.032)	.926(.012)
AVERAGE	.878	.892	.930	.917

TABLE I: Average hiding time of evader as a percentage of the total simulation time. Changing from dense sampling at low frequencies to sparse sampling at high frequencies does not significantly affect the hiding time. Variation is slightly higher for Roadmap 2 (randomly sampled) than for Roadmap 1 (grid cell).

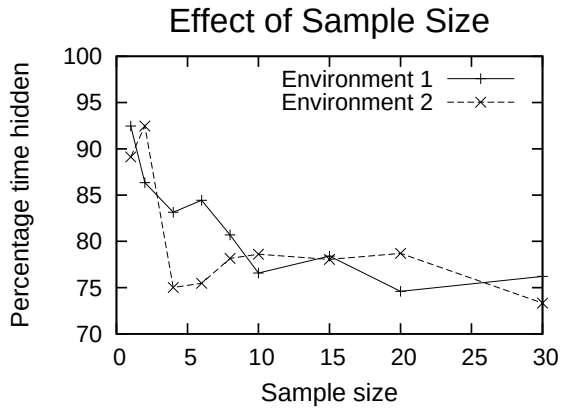


Fig. 4: Effect of sample size on the percentage of time the evader is hidden. A lower percentage corresponds to a more accurate probability model.

C. Evader Model vs. Update Frequency

Unlike other approaches, our method does not require the transition probabilities to be updated at intervals of a fixed size. Rather, it is robust to a range of different update frequencies because in the model of the evader's movement, the evader's velocity is expressed in terms of the amount of time between updates. We verify in this experiment that adjusting the frequency of updates does not significantly affect the accuracy of the model as measured by the percentage of time the evader stays hidden.

For each of the two test environments, we use two roadmaps – a regular grid and one made using MAPRM [9], a roadmap whose nodes lie on the medial axis of the free space. Both roadmaps have the same number of nodes and therefore result in the same number of partitions. We keep the average number of samples per timestep constant by increasing the number of samples per update as we decrease the frequency of updates. This is done in order to isolate the effect of update frequency. We start at 5 samples per partition per update, with an update period (time between successive updates) of 10 timesteps, and increase the update period gradually to 60 timesteps.

Our results are shown in Table I. In both environments, our results indicate that calculating transition probabilities at low frequencies does not produce significantly different performance than doing so at high frequencies. This validates our approach of adjusting the evader's movement model to account for the update frequency.

D. Capture time for different roadmaps

Unlike prior approaches, the method presented in this paper does not require the environment to be partitioned in one particular way in order to work.[4] Consequently, we can partition the environment in any manner that is convenient or advantageous. The following experiment demonstrates why being able to do so is important, by highlighting an example in which the particular partitioning affects the ability of the pursuer to capture the evader.

Since the pursuer uses the roadmap for planning motions in addition to tracking the evader, using a significantly different roadmap will not only result in a different partitioning of the environment but also cause the pursuer to move through the environment differently. We show that this affects the pursuer's success rate; in particular, we hypothesize that using a randomized roadmap is likely to lead to a higher capture rate than using an axis-aligned grid cell roadmap because the former makes the pursuer's angle of approach less predictable.

In this experiment, we compare the number of runs that end in capture for two different partitions of the same environment. We test this for the same two environments as in the previous experiment, using the same two roadmaps. The simulation is ended either when the evader is captured or after 10000 timesteps.

Our results for this experiment, presented in Table II, are interesting. The results on the office-like environment strongly support our hypothesis: the capture rate when using the randomized roadmap is three times higher than when using the grid cell roadmap. In the rocky environment, superficially the results appear to disagree with the hypothesis: there are no captures at all using the randomized roadmap but 6 using the grid cell roadmap. However, a close look at

[4]This does not necessarily mean the tracker will cause the pursuer to move through the environment in the same way for different partitionings, since different partitionings may contain partitions that are different in number, size and shape. Rather, we merely claim that the environment need not be partitioned in one particular way in order for the algorithm to work.

the results reveals that all 6 captures occur with the first 60 timesteps. This is because the initial positions of the pursuer and evader vary slightly for each run due to the different random seeds. In some runs, their positions make it impossible for the evader to escape since it is already cornered. 60 timesteps corresponds to the time taken for the pursuer to intercept the evader.

If we discount these outliers, then we still need to explain why the randomized roadmap outperforms the grid cell for the office environment but both perform poorly in the rocky environment. The answer is in the shapes and placement of the obstacles. In the office environment, obstacles are axis-aligned with the grid cells, resulting in numerous right-angled corners. In such an environment, the randomness of the second roadmap is advantageous to the pursuer because it could approach at an angle oblique to the corners, blocking off a clear escape path for the evader. In the rocky environment, there are fewer corners in which the evader could be trapped in this manner. Regardless of the angle of approach, the evader is always able to escape since it is faster.

Despite mixed results with respect to our hypothesis on which roadmap leads to a higher capture rate, the experiment nevertheless successfully demonstrates that the partitioning affects the pursuer's movements and success.

Captures			
Env.1 (62 nodes)		Env.2 (114 nodes)	
Roadmap 1	Roadmap 2	Roadmap 1	Roadmap 2
3(.1)	13(.43)	6(.2)	0(.0)

TABLE II: Captures and capture rates for both environments, using different roadmaps. In the office-like environment (Env.1), using a roadmap with randomness significantly improves the capture rate. In the second environment, using a grid-cell leads to higher capture rates than using a randomized roadmap, but both rates are lower than in the first environment. This is because there are fewer corners in which the evader could be trapped, giving it more options to escape after being sighted.

VI. CONCLUSIONS

In this paper, we introduced a method for estimating transition probabilities for adjacent partitions in an arbitrarily partitioned environment. This allows us to generalize the probabilistic formulation of the pursuit evasion problem to arbitrary partitions, and by extension to arbitrary 2D environments. We adapted this method to probabilistic roadmaps by creating Voronoi diagrams using the roadmap nodes as sites. Our method is online and robust to the number of timesteps between updates and the capabilities of the evader. It runs efficiently, allows us to consider a variety of agent capabilities and environments, and illustrates the utility of probabilistic roadmaps for pursuit evasion.

We are excited by the possibility of extending the approach to higher-dimensional environments. We would also like to consider dynamic adjustment of the time between updates to the transition probabilities according to the pursuers' positions and *a priori* probabilities. For example, we may wish to bias sampling towards partitions that are nearer to the pursuers or that have *a priori* high probabilities.

REFERENCES

[1] T. Parsons, *Pursuit-Evasion in a Graph*. Springer-Verlag, 1978, pp. 426–441.

[2] L. J. Guibas, J. claude Latombe, S. M. Lavalle, D. Lin, and R. Motwani, "Visibility-based pursuit-evasion in a polygonal environment," in *International Journal of Computational Geometry and Applications*. Springer-Verlag, 1997, pp. 17–30.

[3] S. Rodriguez, J. Denny, T. Zourntos, and N.M.Amato, "Toward simulating realistic pursuit-evasion using a roadmap-based approach," in *Lecture Notes in Computer Science*, 2010, pp. 82–93.

[4] S. Rodriguez, J. Denny, A. Mahadevan, J. C.-T. Vu, J. Burgos, T. Zourntos, and N. M. Amato, "Roadmap-based pursuit-evasion in 3d structures," in *Proc. Int. Conf. Computer Anim. Social Agents (CASA)*, 2011.

[5] S. Rodriguez, J. Denny, , J. Burgos, A. Mahadevan, K. Manavi, L. Murray, A. Kodochygov, T. Zourntos, and N. M. Amato, "Toward realistic pursuit-evasion using a roadmap-based approach," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2011.

[6] D. Hsu, L. E. Kavraki, J.-C. Latombe, R. Motwani, and S. Sorkin, "On finding narrow passages with probabilistic roadmap planners," in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, 1998, pp. 141–153.

[7] D. Hsu, J.-C. Latombe, and H. Kurniawati, "Foundations of probabilistic roadmap planning," in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2006.

[8] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo, "OBPRM: An obstacle-based PRM for 3D workspaces," in *Robotics: The Algorithmic Perspective*. Natick, MA: A.K. Peters, 1998, pp. 155–168, proc. Third Workshop on Algorithmic Foundations of Robotics (WAFR), Houston, TX, 1998.

[9] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, "MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 2, 1999, pp. 1024–1031.

[10] R. Vidal, O. Shakernia, H. Kim, D. H. Shim, and S. Sastry, "Probabilistic pursuit-evasion games: Theory, implementation, and experimental evaluation," in *IEEE Transaction son Robotics and Automation*, 2002, pp. 662–669.

[11] J. P. Hespanha, M. Prandini, and S. Sastry, "Probabilitstic pursuit-evasion games: A one-step nash approach," in *Proceedings of the IEEE Conference on Decision and Control*, 2000, pp. 2272–2277.

[12] G. Hollinger, A. Kehagias, and S. Singh, "Probabilistic strategies for pursuit in cluttered environments with multiple robots," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2007, pp. 3870–3876.

[13] D. Hsu, W. Lee, and N. Rong, "A point-based pomdp planner for target tracking," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2008, pp. 2644–2650.

[14] A. Kolling and S. Carpin, "Pursuit-evasion on trees by robot teams," *Trans. Rob.*, vol. 26, no. 1, pp. 32–47, 2010.

[15] S. M. Lavalle, D. Lin, L. J. Guibas, J. claude Latombe, and R. Motwani, "Finding an unpredictable target in a workspace with obstacles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1997, pp. 737–742.

[16] V. Isler, S. Kannan, and S. Khanna, "Randomized pursuit-evasion with limited visibility," in *Proc. ACM-SIAM Symposium on Discrete Algorithms*, 2004, pp. 1060–1069.

[17] B. P. Gerkey, S. Thrun, and G. Gordon, "Visibility-based pursuit-evasion with limited field of view," *Int. J. Robot. Res.*, vol. 25, no. 4, pp. 299–315, 2006.

[18] H. Kim, R. Vidal, D. Shim, O. Shakernia, and S. Sastry, "A hierarchical approach to probabilistic pursuit-evasion games with unmanned ground and aerial vehicles," in *Proc. IEEE Conf. on Decision and Control*, 2001, pp. 634–639.

[19] A. Kolling, A. Kleiner, M. Lewis, and K. Sycara, "Solving pursuit-evasion problems on height maps," in *IEEE International Conference on Robotics and Automation (ICRA 2010) Workshop: Search and Pursuit/Evasion in the Physical World: Efficiency, Scalability, and Guarantees*, 2010.