Path-Guided Artificial Potential Fields with Stochastic Reachable Sets for Motion Planning in Highly Dynamic Environments

Hao-Tien Chiang[1], Nick Malone[1], Kendra Lesser[2], Meeko Oishi[2], Lydia Tapia[1]

***Abstract*— Highly dynamic environments pose a particular challenge for motion planning due to the need for constant evaluation or validation of plans. However, due to the wide range of applications, an algorithm to safely plan in the presence of moving obstacles is required. In this paper, we propose a novel technique that provides computationally efficient planning solutions in environments with static obstacles and several dynamic obstacles with stochastic motions. Path-Guided APF-SR works by first applying a sampling-based technique to identify a valid, collision-free path in the presence of static obstacles. Then, an artificial potential field planning method is used to safely navigate through the moving obstacles using the path as an attractive intermediate goal bias. In order to improve the safety of the artificial potential field, repulsive potential fields around moving obstacles are calculated with stochastic reachable sets, a method previously shown to significantly improve planning success in highly dynamic environments. We show that Path-Guided APF-SR outperforms other methods that have high planning success in environments with 300 stochastically moving obstacles. Furthermore, planning is achievable in environments in which previously developed methods have failed.**

I. Introduction

Motion planning consists of finding a valid, collision-free path for a robot from a start configuration to a goal configuration. Planning with both static and dynamic obstacles is complicated by the need for constant adjustments of plans to account for moving obstacles, yet critical in applications such as flight coordination and autonomous vehicles. In these dynamic environments, it is important to produce trajectories that avoid both static and dynamic obstacles with high success rates in a computationally efficient manner.

Sampling based methods have shown great success at identifying valid paths in complex planning problems with static obstacles [1], [2], [3]. We extended this work to accommodate dynamic obstacles [4], and recently showed that an Artificial Potential Field (APF) method outperformed these sampling-based methods in highly dynamic environments [5]. APF works by following the combined gradient of repulsive potentials around obstacles and an attractive potential from the goal. However, like all APF methods, this approach struggles with the local minima problem. While there are a few extensions that address this known limitation [6], [7], they are either computationally expensive or perform badly in some environments. Researchers have also begun to explore combinations of sampling based methods and APF in swarm robotics [8] and group behavior simulations [9], to handle the very high dimensional combined configuration space.

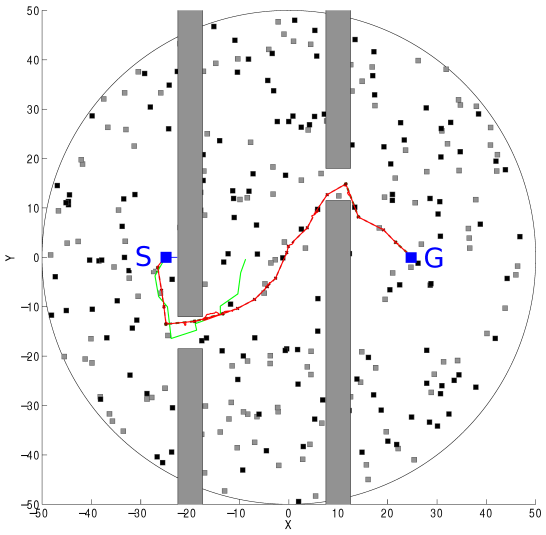


Fig. 1: Narrow environment used for experiments: grey bars are static obstacles, black/grey squares are 300 stochastic moving obstacles in line/arc trajectories, and the robot must traverse from S (start) to G (goal). Solid red line is the actual path taken by Path-Guided APF-SR initially guided by the PRM path, dashed line. For comparison, the blue line is the path taken by APF-SR (trapped in local minima near start) and the green line is the path of SR-Query (collision).

In this paper, we propose the Path-Guided APF-SR an alternative approach that has low computational run time cost, works flexibly with any path generating sampling method, and shows great success at avoiding collisions in crowded environments with hundreds of moving obstacles. We focus on the problem of safe navigation for a single robot in highly dynamic and complex environments, with both static and dynamic, stochastic obstacles (which are not controllable). The main contributions of this work are 1) the combination of path generation via sampling methods and dynamic, stochastic obstacle avoidance via APF methods weighted by stochastic reachable sets, 2) effective navigation in environments with both static obstacles as well as dynamic, stochastic obstacles, 3) a path-guided gradient for APF, and 4) evaluation of our method in multiple environments and

[1]Computer Science, University of New Mexico, Albuquerque, NM 87131
[2]Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131

with multiple sampling methods.

Our method works by first applying a sampling-based technique to identify a path that's collision-free with respect to static obstacles. Then, an artificial potential field planning method is used to safely navigate through the moving obstacles, using the path as an attractive intermediate goal bias. To improve the safety of the artificial potential field, we incorporate a repulsive potential field for each moving obstacle, based on pre-computed stochastic reachable (SR) sets. These sets provide a pairwise assurance of a likelihood of safety, for known relative robot-obstacle stochastic dynamics, at exponential computational cost in the number of relative degrees of freedom. The APF-SR combination method removes this exponential cost from runtime, while still maintaining high planning success rates in highly dynamic environments [5].

We compare Path-Guided APF-SR to two algorithms we previously developed for navigation in environments with dynamic, stochastic obstacles (but without any static obstacles). First, SR-Query [4] is a PRM-based method that dynamically adjusts path selection based on SR set weighted edges that are updated as obstacles move. This method has been shown to outperform a common PRM-based method [10] for dynamic obstacle avoidance. Second, APF-SR [5] is a SR-biased APF method that showed a higher success rate than standard APF methods for avoiding large numbers of highly dynamic obstacles. In addition, we compare Path-Guided APF-SR to ORCA [11], an algorithm developed for multi-agent path finding using a predicted region of collision, velocity obstacles.

We tested Path-Guided APF-SR in environments with 300 stochastic moving obstacles, that either are free of static obstacles, or have challenging static obstacles in the shape of a "bug trap" or narrow corridors. In these tests, our method avoids local minima and has a high success rate (over 90%) in highly dynamic environments. Further, our method is capable of solving problems that are not solvable by APF-SR or SR-Query. We also evaluated Path-Guided APF-SR with paths from two different sampling methods, PRM and EST, and showed no significant change in success rate. The enclosed video submission contains the experiments and visualization of the simulations.

II. Related Work

APF [12] is a powerful approach to path planning, that is simple to construct, suitably fast for online planning, easily handles kinodynamic and nonholonomic constraints, and applicable to problems in unmanned aerial vehicles [13], [14], robot soccer [15], and mobile robots [16], [17], [18], [19]. APF methods have been extended to address problems in which the goal is not reachable due to obstacle proximity [16], and navigation in narrow passages is required [17]. Other recent work has focused on modification of the computation of the potential field through fuzzy [19] and evolutionary [18] APFs. APFs based on the repulsive and attractive concepts can integrate with other path planning methods [20], [21]. In [20], a user defined costmap influences node placement in a Rapidly exploring Random Tree (RRT) algorithm. The costmap dictates a repulsiveness or attractiveness factor for every region. Similarly, navigation fields [21] assign a gradient which agents follow, with application to crowd modeling. A method based on concept of affordance for multi-agent planning was proposed in [22].

The main drawback in APF methods is the possibility of becoming trapped in local minima. While several approaches have been proposed, this remains a difficult problem. The Randomized Path Planner [6] prescribes a random walk when at a local minimum, which while effective in some environments, often requires a long time to escape from complex environments (e.g., a bug trap with a long and narrow escape route). The navigation function approach [7] is computationally expensive and often restrictive.

Path-guided APF has been applied to swarm [8] or group [9] robotics by generating the attractive potential from intermediate goals along a precomputed path, allowing a large number of robots to efficiently navigate in complex environments in a coordinated way. Path-Guided APF-SR focuses on safely navigating a single robot in a complex environment with many uncontrollable stochastically moving obstacles. This requires a new path-guidance scheme to avoid local minima and a SR-biased repulsive potential to improve safety.

Reachability analysis has been used to inform motion planning decisions for collision avoidance, based on two approaches. One approach allows for a control and disturbance input, and generates the maximal set of initial states within which a collision is guaranteed, assuming the worst case disturbance input [23], [24]. This set, known as the reachable set, can be obtained by solving a Hamilton-Jacobi-Isaacs (HJI) equation [25]; its complement assures collision avoidance. In [26], the reachable set (for the robot to reach target while avoiding a single obstacle) is computed by assuming the obstacle actively attempts to collide with the robot. A similar approach is used in [27], with reachable sets computed iteratively to enable the robot to modify its actions. Multiple obstacles are avoided in an online fashion in [28], based on precomputed invariant sets.

Another approach incorporates stochastic relative dynamics. Probabilistic SR sets [29] describe the set of states in which a collision is guaranteed with a certain probability. In [30], the obstacles are modeled as random sets that evolve over time, whereas the desired target set is known and unchanging. A disturbance input that acts in opposition to the robot's objective is considered in [31], and the reachable sets are generated using a two-player game formulation.

III. Preliminaries

A. Obstacle Dynamics

We consider 2D rigid body obstacles in the shape of a square, with center of mass $\overline{x}^o = (x^o, y^o)$. We presume that the obstacles follow trajectories described by a straight line or a constant radius arcs, with stochastic linear or angular velocity $w \in \mathcal{W}$, respectively, a discrete random variable with probability distribution $p(w)$. However, more complex dynamics, e.g., ones that switch between straight line and

constant radius arc movements, can easily be incorporated. The obstacle dynamics are discretized via an Euler approximation with time step Δ.

$$\overline{x}^o_{n+1} = \overline{x}^o_n + f^o(w_n, n)\Delta \tag{1}$$

For the straight-line trajectories with line slope $\alpha \in \mathbb{R}$,

$$f^o(w_n, n) = \begin{bmatrix} w_n \\ \alpha w_n \end{bmatrix} \tag{2}$$

For the constant-arc trajectories with radius $r \in \mathbb{R}^+$,

$$f^o(w_n, n) = \begin{bmatrix} r\left(\cos(w_n(n+1)) - \cos(w_n n)\right) \\ r\left(\sin(w_n(n+1)) - \sin(w_n n)\right) \end{bmatrix} \tag{3}$$

B. Relative robot-obstacle dynamics

We presume the robot is a point mass with dynamics

$$\overline{x}^r_{n+1} = \overline{x}^r_n + f^r(u_n)\Delta, \tag{4}$$

representative of either 1) a holonomic system with state $\overline{x}^r = (x^r, y^r)$, or 2) a non-holonomic system with state $\overline{x}^r = (x^r, y^r, \theta^r)$. For the holonomic system,

$$f^r(u_n) = u \tag{5}$$

with velocity control input $u = (u^x, u^y) \in \mathcal{U} \subseteq \mathbb{R}^2$, and for the nonholonomic system,

$$f^r(u_n) = \begin{bmatrix} u^s_n \cos(\theta^r_n) \\ u^s_n \sin(\theta^r_n) \\ u^w_n \end{bmatrix} \tag{6}$$

with linear and angular velocity inputs $u = (u^s, u^w) \in \mathcal{U}$.

A collision occurs when $\|\overline{x}^r_n - \overline{x}^o_n\|_2 \leq d_{\text{obs}}$, or equivalently in relative coordinates $\tilde{x} \equiv \overline{x}^r - \overline{x}^o \in \mathcal{X}$, when

$$\|\tilde{x}_n\|_2 \leq d_{\text{obs}} \tag{7}$$

Relative dynamics are described in relative coordinates by

$$\tilde{x}_{n+1} = \tilde{x}_n + [f^r(u_n) - f^o(w_n, n)]\Delta \tag{8}$$

The stochastic transition kernel $\tau(\tilde{x}_{n+1} \mid \tilde{x}_n, u_n, n)$ represents the probability distribution of $\tilde{x}_{n+1}$ conditioned on the known values $\tilde{x}_n$, u_n at time step n.

C. SR Sets for Collision Avoidance

We calculate collision avoidance probabilities by solving a stochastic reachability problem with the avoid set, $\overline{K}$, defined as the set of states in which a collision is said to occur (7). We compute the probability that the robot remains within K, the complement of $\overline{K}$, over N time steps, with initial relative position $\tilde{x}_0$, using dynamic programming [32], [29]. We summarize key elements of the derivation here, and refer the reader to [4] [5] for additional details.

As in [29], the SR set is generated by iterating the value function backwards in time with $V_N(\tilde{x}) = \mathbf{1}_K(\tilde{x})$, and

$$\begin{aligned} V_n(\tilde{x}) &= \mathbf{1}_K(\tilde{x}) \int_{\mathcal{X}} V_{n+1}(\tilde{x}')\tau(\tilde{x}' \mid \tilde{x}, u, n)\, d\tilde{x}' \qquad (9) \\ &= \mathbf{1}_K(\tilde{x}) \sum_{w \in \mathcal{W}} V^*_{n+1}\left(\tilde{x} + \Delta f^r(u, \theta)\right. \\ &\qquad \left. -\Delta f^o(w, n)\right) p(w). \qquad (10) \end{aligned}$$

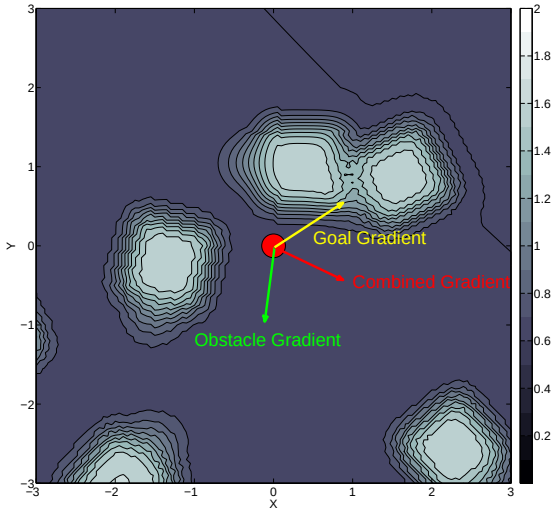


Fig. 2: Potential landscape gradients: Repulsive APFs (square-like contours) are queried from SR sets convolved with a Gaussian ($\sigma = 0.15$). The gradient of the repulsive APF is shown in the green arrow. The attractive APF is weaker and has a gradient (yellow arrow) that points toward the goal located at the upper right corner.

from time $n = N$ to time $n = 0$. The indicator function $\mathbf{1}_K(x)$ is equal to 1 if $x \in K$ and equal to 0 otherwise. The value function $V^*_0(\tilde{x}_0)$ at time $n = 0$ describes the probability of avoiding collision over N time steps from an initial state $\tilde{x}_0$. The optimal control input u to avoid collision is the control that solves

$$V^*_n(\tilde{x}) = \max_{u \in \mathcal{U}} \left\{ \mathbf{1}_K(\tilde{x}) \sum_{w \in \mathcal{W}} V^*_{n+1}\left(\tilde{x} + \Delta f^r(u, n)\right. \right. \\ \left. \left. -\Delta f^o(w, n)\right) p(w) \right\}. \tag{11}$$

IV. Methods

Path-Guided APF-SR separates planning into an offline phase and a run time phase. In the offline phase, a sampling-based method is used to generate a path to the goal in the presence of static obstacles only. Then, this path is used during the run time phase as a heuristic to guide the APF-SR method (which considers the moving obstacles). This eliminates the local minimum created by static obstacles and retains the high success rate of APF-SR. Hence, Path-Guided APF-SR is capable of fast real-time planning in environments with a large number of moving obstacles.

A. Offline Phase: Computation

Path-Guided APF-SR requires a pre-computed stochastic reachable set and a path. Since the stochastic reachable set (11) can have discontinuities and plateaus, which are detrimental to a gradient-following APF method, we convolve the

Algorithm 1 PRM-Guided-APF-SR

Input: Moving obstacles O_M with pre-computed smoothed SR sets, static obstacles O_S, sampling-based $path$, robot r start configuration $\mathcal{S}$ and goal configuration $\mathcal{G}$

1: $V_{nextnode} = \mathcal{S}$
2: **for** $t = 0$; $t < maxTime$; $t = t + \Delta$ **do**
3: **for** Obstacle $o \in O_M$ **do**
4: updateObstacle(t,o,$o.w$,$o.p(w)$)
5: **end for**
6: $APF_{gradient} = (0, 0)$
7: **for** Obstacle $o \in O_M$ **do**
8: **if** dist($\overline{x}_n^o$, $\overline{x}_n^r$) $< d_{min}$ **then**
9: $APF_{gradient} =$
10: $APF_{gradient} + o.queryAPFGradient(\overline{x}_n^r)$
11: **end if**
12: **end for**
13: **for** Obstacle $o \in O_S$ **do**
14: $APF_{gradient} =$
15: $APF_{gradient} + o.calcAPFGradient(\overline{x}_n^r)$
16: **end for**
17: **if** $dist(\overline{x}_n^r, V_{nextnode}) < \epsilon$ **then**
18: $(V_{nextnode}, \mathcal{P}) = getNewTarget(\overline{x}_n^r, path, \mathcal{G})$
19: **end if**
20: $APF_{gradient} =$
21: $APF_{gradient}$+$getPathGuidedGradient(V_{nextnode}, \mathcal{P})$
22: $u = getControl(APF_{gradient})$
23: $\overline{x}_{n+1}^r = \overline{x}_n^r + \Delta \cdot f^r(u, t)$
24: **if** dist($\overline{x}_n^r$, $\mathcal{G}$) $< Goal_{threshold}$ **then**
25: break
26: **end if**
27: **end for**

stochastic reachable set with a 2D Gaussian of width σ for smoothing (Figure 2).

Next, the required path is extracted from a sampling-based method such as PRM [2], RRT [33] or EST [1] which only considers the static obstacles. In principle, any sampling-based method can be used, provided the regions containing the start and goal configurations are connected.

B. Run Time Phase: Planning

Path-Guided APF-SR (Algorithm 1) first updates the obstacle positions (line 3-5, $updateObstacle$). The repulsive potential has two primary components, due to moving and static obstacles. The moving obstacle repulsive potential is queried by using the pre-computed SR sets (line 7-11, $queryAPFGradient$). The static obstacle repulsive potential is computed (line 12-14, $calcAPFGradient$). The attractive potential is computed by finding the next node on the path which is required to generate a path-guided potential (lines 15-17, $getNewTarget$ and $getPathGuidedGradient$). The attractive and repulsive potentials are combined to compute and execute the robot's action (line 19-20, $getControl$). The subroutines are described in more detail below.

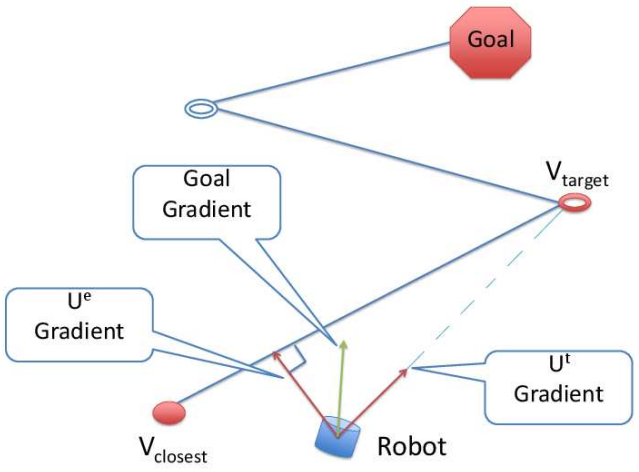


Fig. 3: Path-Guided APF: The Path-Guided Gradient is calculated by adding the Edge Following Gradient (perpendicular to the edge) and the Next Node Following Gradient (points toward the next node). The Path-Guided Gradient guides the robot to follow the path generated by sampling-based methods.

Subroutine $updateObstacle$ moves the obstacle with its current velocity or angular velocity. Velocities are sampled every interval T from a set of velocities $\mathcal{W}$ identical to the offline SR set calculation.

For every moving obstacle within a distance, d_{min}, from the robot, $queryAPFGradient$ computes the repulsive APF gradient by finding the relative position from the obstacle to the robot then querying the smoothed SR set collision probability for the APF value (Section III). The gradient is then calculated by the second order central difference method and summed into the vector $APF_{gradient}$ for each obstacle.

Subroutine $calcAPFGradient$ calculates the gradient of a repulsive APF (U_S) generated by a static obstacle. U_S can be calculated in the fashion similar to [12] based on the distance d to the obstacle. To implement the same APF decay around both static and moving obstacles, we use:

$$U_S = \begin{cases} erfc(\frac{d}{\sqrt{2}\sigma}) & \text{if } d^* \geq d \geq 0 \\ 0 & \text{if } d > d^* \end{cases}$$

The convolution of a Gaussian with variance σ and a square of width w, where $w \gg \sigma$, results in an complementary error function ($erfc$) type decay on all four edges. Given a small σ (e.g., compared to the obstacle size), the moving and static obstacles generate comparable APFs.

In order to compute the attractive potential, subroutine $getNewTarget$ finds the next node in the path. With a PRM, the path is found by running Dijkstra's algorithm. For other methods, such as a tree-based method, there is a single path, so the next path node is simply used. As shown in Figure 3, subroutine $getPathGuidedGradient$ computes the Edge Following Gradient and the Next Node Following gradient. The Edge Following Gradient is a unit vector perpendicular to the edge that points toward the edge. The Next Node Following Gradient is a unit vector that

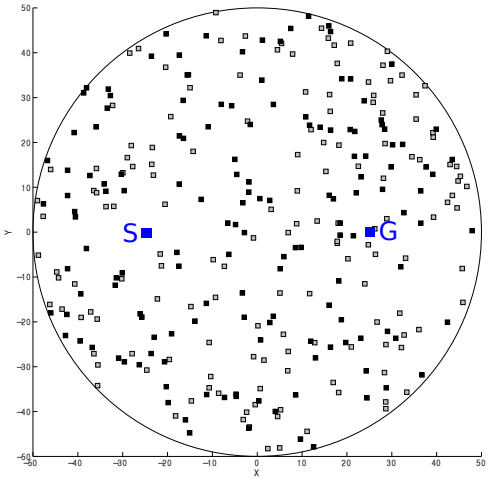


Fig. 4: Obstacle-free environment in which the robot must traverse from S (start) to G (goal). The black/grey squares represent stochastic linear/arc moving obstacles.

points toward the next node $V_{nextnode}$. The sum of these two gradients forms the Path-Guided Gradient. This gradient pulls the robot toward the path as close as possible, with deviations from the path due to moving obstacles.

Finally, the $getControl$ subroutine finds the control closest to the combined gradient of all obstacles and goal, ($APF_{gradient}$). For example, in the case of a holonomic robot, the control input vector u is collinear with $APF_{gradient}$. On the other hand, for nonholonomic, e.g., unicycle robots, $getControl$ returns control inputs that attempt to turn the robot toward $APF_{gradient}$ at the max turn rate and accelerate/decelerate the robot so that the inner product of u and $APF_{gradient}$ is maximized.

V. Experiments

To demonstrate the Path-Guided APF-SR algorithm, we tested three different environments with stochastic, dynamic obstacles: The free environment has no static obstacles, similar to [4] and [5]. The bug trap environment has a static obstacle the robot must escape to reach the goal. In the narrow corridor environment, the robot must pass through two narrow openings to reach the goal.

We created ten PRM roadmaps with n=1000 nodes and edges selected by connecting nodes to their k=10 nearest neighbor connections for each of the three static environments. Ten trials were performed on each roadmap. We evaluated metrics including success rate and path length for each trial. We evaluated the efficacy of our algorithm for both holonomic (5) and non-holonomic (6) robot dynamics. The robot input is constrained by a maximum linear velocity of 0.36 units per second and a maximum angular velocity of $\pi/5$ radians per second.

At runtime, the robot replans every Δ seconds. The environment has 300 randomly placed moving obstacles, with 150 following straight-line dynamics (2), and 150 following constant-arc dynamics (3). The moving obstacles are squares with width $d_{obs} = 1$. Smoothed SR sets are computed for each obstacle with $\sigma = 0.15$; for each of the straight-line obstacles, $\alpha \in [0, 2\pi)$ was selected randomly. The set of possible linear velocities is $\mathcal{W} = \{0.1, 0.2, 0.5, 0.7\}$ with probability $p(w) = \{0.3, 0.2, 0.3, 0.2\}$; the set of possible angular velocities is $\mathcal{W} = \{0.17, 0.26, 0.39, 0.52\}/r$ with probability $p(w) = \{0.2, 0.2, 0.3, 0.3\}$, with $r \in \{5, 10, 15\}$.

To maintain the constant density of moving obstacles, we restrict the robot and moving obstacles to lie in a circle with radius 50. When an obstacle hits the boundary of the circle, it is transported to the antipodal position on the circle and continues evolving from this new position. The resulting density of moving obstacles is similar to that in [5]. We presume complete information, that is, that the robot has access to all obstacle positions within radius $d_{min} = 3$ units.

We note that the relative strength between obstacle potential gradient and goal potential gradient affects the success rate and path length. If the goal gradient is too strong, the robot collides with moving obstacles more often. If the goal gradient is too weak, the robot focuses too much on avoiding obstacles and often takes a much longer path to reach the goal. We empirically select a ratio of 1:100 for the weighting between the goal gradient and the obstacle gradient, to balance these potentially competing objectives.

We compare our method to SR-Query [4], APF-SR [5] modified with a standard APF static obstacle repulsion, ORCA [11] downloaded from [34] and modified for a single robot and moving obstacles. All experiments are implemented with MATLAB on an i7-3615QM with 16GB RAM.

A. Obstacle-Free Environment

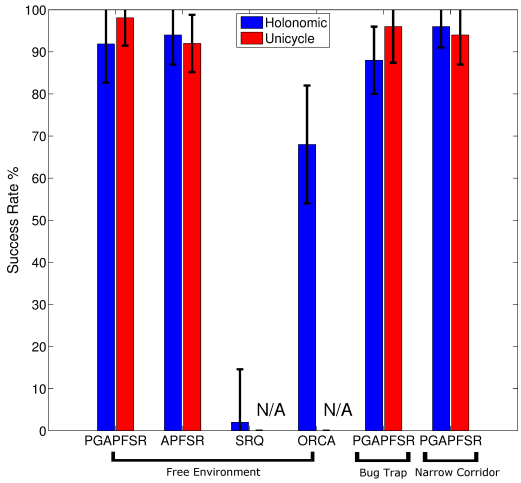
The obstacle-free environment (Figure 4) provides a baseline for direct comparison of Path-Guided APF-SR with APF-SR, and demonstrates the PRM guidance does not interfere with APF-SR moving obstacle avoidance. The robot starts at $(-25, 0)$ and the goal is at $(25, 0)$.

Figure 5a shows that Path-Guided APF-SR has a success rate comparable to APF-SR ($> 90\%$), and a comparable path length. In contrast, SR-Query has a mere 2% success rate, due to collisions the robot could not avoid when traveling along an edge. When the number of moving obstacles is significantly reduced to 50 we observed a success rate of 58%, consistent with that reported in [4]. Although ORCA produces the shortest path length, it does not have a high success rate (68%). Unlike the stochastic reachability analysis, the stochastically changing obstacle speed was not considered in ORCA's calculations.

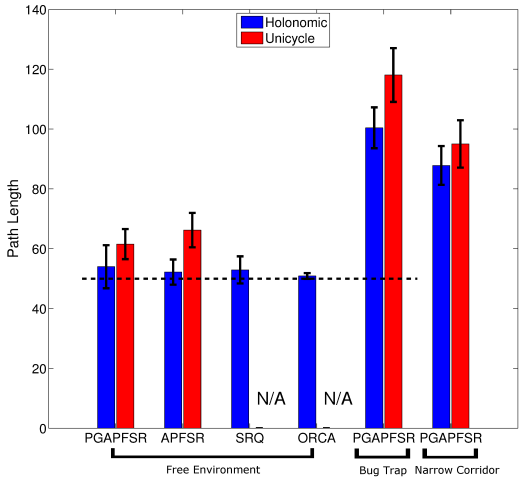
B. Bug Trap Environment

The bug trap environment (Figure 6) is designed to test the algorithm's ability to avoid local minima created by static obstacles. The start and goal positions are the same as in the obstacle-free environment, and the obstacle width is 5 times the width of the moving obstacles.

Figure 5a demonstrates the utility of the PRM for path guidance. While the success rate for Path-Guided APF-SR



(a) Success rate



(b) Path length

Fig. 5: Performance comparison: (a) Success rate of Path-Guided APF-SR (PGAPFSR in the figure) compared to APF-SR (APFSR), SR-Query (SRQ) and ORCA in various environments. SR-Query and ORCA are unable to directly handle unicycle dynamics, and the success rate of the former in the bug trap and narrow corridor environment is less than 1%. APF-SR became trapped in local minima in the bug trap and narrow corridor environment and never reached the goal. (b) Path length of various methods. The dotted line represents the straight line distance from start to goal.

is comparable to that in the obstacle-free environment (for both holonomic and non-holonomic robot dynamics), APF-SR became trapped in a local minimum created by the static obstacles and the goal. While SR-Query avoids the local minima problem as it is not an APF-based method, the longer path required in this environment increases the probability of collision while the robot travels along an edge. This results in zero success in 100 trials.

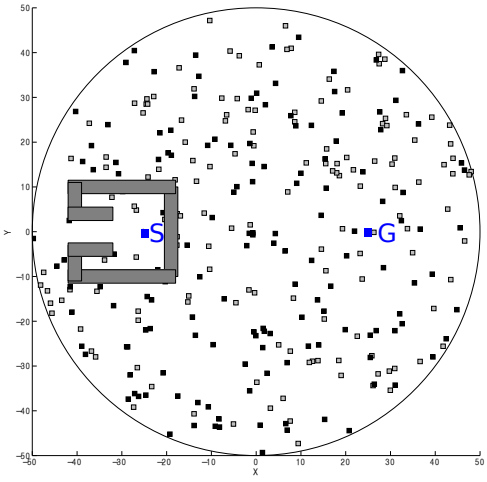


Fig. 6: Bug trap environment in which the robot must traverse from S (start) to G (goal). Grey bars are static obstacles; black squares are 300 stochastic, dynamic obstacles.

Environment	PRM-APF-SR	APF-SR	SR-Query
Free	3.3 ms	3.0 ms	84.0 ms
Bug Trap	9.7 ms	6.6 ms	57.0 ms
Narrow Corridor	8.8 ms	5.1 ms	101.0 ms

TABLE I: Run time per planning step

C. Narrow Corridor Environment

The narrow corridor environment (Figure 1a) is designed to test the algorithm in environments challenging to sampling-based planners. The openings in the narrow corridors are fives times as wide as the moving obstacles.

Path-Guided APF-SR has a high success rate despite stochastic and static obstacles. Although the environment does not contain deep potential minima like the bug trap environment, APF-SR still becomes trapped in local minima. SR-Query was unsuccessful in all 100 trials, since the longer path required in this environment increases the probability of collision while the robot travels along an edge.

D. Discussion

The bug trap and narrow corridor environment clearly show Path-Guided APF-SR can avoid local minimum created by static obstacles and also successfully navigate around stochastic dynamic obstacles. It is a superior method to both APF-SR and SR-Query. Most importantly (Table I), the run time per planning step is similar to APF-SR, and is at least one order of magnitude less than SR-Query, making it feasible for applications that require real-time online planning. Also, considering the stochastic motion helps it outperform ORCA.

The primary cause of failure for Path-Guided APF-SR seems to be multiple moving obstacles interaction. Consider the case in which several obstacles converge on the robot from different directions. The obstacle potential is queried from SR sets that consider interaction with a single obstacle

Roadmap	PRM-APF-SR	SR-Query
$n = 1000$, $k = 10$	94%	2%
$n = 250$, $k = 6$	94%	0%

TABLE II: Comparison of planning success given paths extracted from PRMs of various sizes in the free environment. Size of roadmap is defined by n nodes, and k nearest neighbors selected for edge connection.

in isolation. The presence of multiple obstacles may create local minima that lead to collisions. Unfortunately, a multi-obstacle SR set is not practical since computation is precluded by the high dimensionality of the relative state space. However, the occurrence of such an event is rare compared to single obstacle encounters, and therefore does not have a large effect on the success rate. We investigated the use of the path found by SR-Query to guide the APF-SR robot away from regions with high densities of moving obstacles. Although SR-Query can lead the robot to lower density regions, this does not necessarily preclude the multi-obstacle scenario, and in general results in a much longer path length and run time per planning step.

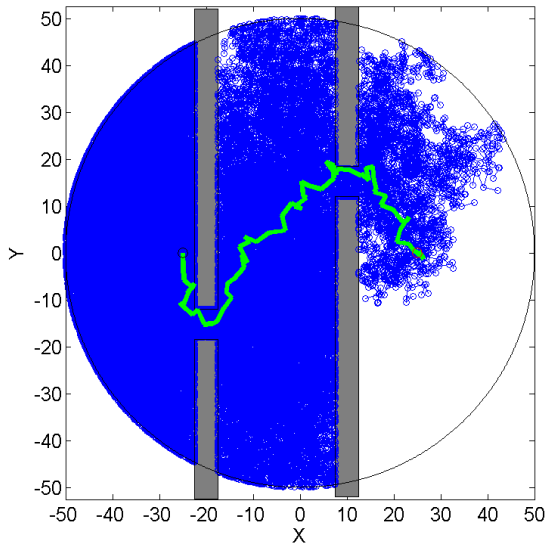
We did not observe any collisions between the robot and the static obstacles using the path-guidance. This is expected, as stochastically moving obstacle avoidance is intrinsically harder than static obstacle avoidance. However, in preliminary experiments using only the next-node following gradient, moving obstacle avoidance occasionally caused large deviations from the guidance path that trapped the robot in a local minimum (results not shown).

Roadmap quality (Table II) has an effect on the success rate of SR-Query, since longer edges increases the probability of collision while traveling on an edge. Path-Guided APF-SR is robust to roadmap quality, so long as it is possible to connect the start position to the goal position.

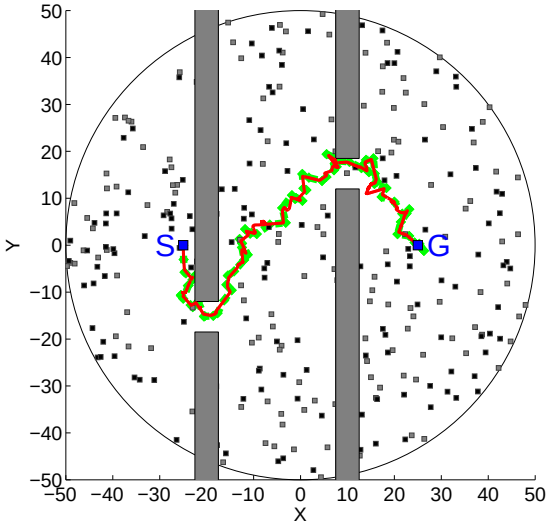
Path-Guided APF-SR can be used with sampling-based methods besides PRM. We implement an Expansive Space Tree (EST) (Figure 7) with 50000 nodes and maximum expansion per edge that is 10 seconds, using with holonomic robot dynamics. While the resulting path is more jagged and about 1.4 times longer than the PRM path shown in Figure 1, the success rate is $92 \pm 6\%$ (e.g., comparable to Path-Guided APF-SRwith PRM). This demonstrates the flexibility of our method, in that it is not restricted to PRM for the offline planner for static obstacles. Indeed, the sampling-based part of our method can be chosen as appropriate for the particular problem at hand.

Further, we believe that our method may also have flexibility in the APF [12], [35], as well. While [5] demonstrates that APF-SR outperforms other choices of APF, it unavoidably inherits the curse of dimensionality in the offline calculation of the SR set. In an environment with a low number of moving obstacles and a scenario where collisions are not fatal, a computationally lightweight APF that can be computed online may be preferable to querying the precomputed SR sets, enabling an online APF planning method for high DOF robot while still avoiding local minima.

Lastly, our method does exhibit the GNRON (Goal Not Reached due to Obstacles Nearby) problem, particularly when navigating through a very narrow corridor. However, techniques such as [16] and [17] can be integrated to alleviate this problem.



(a) EST with 50000 nodes



(b) Actual environment

Fig. 7: Narrow corridor environment, using EST instead of PRM to guide APF-SR (a) EST with 50000 nodes. (b) The dashed line is the path given by EST and the solid line is the actual path taken by the robot.

VI. Conclusion

We propose the Path-Guided APF-SR method for effective navigation in complex environments with both static obstacles as well as numerous dynamic, stochastic obstacles. Integration of PRM for guidance and APF-SR for stochastic dynamic obstacle avoidance enables the robot to bypass local

minima, an often fatal flaw in APF methods. Path-Guided APF-SR has the same success rate and path length as APF-SR, a recently developed method that is highly successful in rich, dynamic, stochastic environments with no static obstacles. Further, the run time is only 10% more than that of APF-SR.

We evaluated Path-Guided APF-SR on static obstacle-free environments, as well as in environments with obstacles designed to target potential weak spots in the algorithm (in both the PRM and APF elements). Our method is successful in the bug trap environment where APF methods typically fail, and also successful in the narrow corridor environment where sampling-based methods would normally be challenged. Further, it is robust to the particular sampling-based planning method, as demonstrated by its comparable success via EST instead of PRM.

VII. Acknowledgements

Chiang, Lesser, and Oishi are supported in part by National Science Foundation (NSF) Grant No. CMMI-1254990 (CAREER Award) and NSF Grant No. CNS-1329878. Tapia and Malone are supported in part by the National Institutes of Health (NIH) Grant P20GM110907 to the Center for Evolutionary and Theoretical Immunology. We would also like to thank Nathanael Rackley for modifying ORCA and running that experiment.

References

[1] Hsu, D., Latombe, J.C., Motwani, R.: Path planning in expansive configuration spaces. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). (1997) 2719–2726

[2] Kavraki, L.E., Švestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Automat. **12**(4) (August 1996) 566–580

[3] LaValle, S.M.: Rapidly-exploring random trees: A new tool for path planning. Technical Report 98-11, Computer Science Dept., Iowa State University (October 1998)

[4] Malone, N., Lesser, K., Oishi, M., Tapia, L.: Stochastic reachability based motion planning for multiple moving obstacle avoidance. In: Hybrid Systems: Computation and Control, HSCC (2014) 51–60

[5] Chiang, H.T., Malone, N., Lesser, K., Oishi, M., Tapia, L.: Aggressive moving obstacle avoidance using a stochastic reachable set based potential fi eld. In: Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR). (2014)

[6] Barraquand, J., Latombe, J.C.: Robot motion planning: A distributed representation approach. Int. J. Robot. Res. **10**(6) (1991) 628–649

[7] Rimon, E., Koditschek, D.: Exact robot navigation using artificial potential functions. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). (1992) 501–518

[8] Alex Wallar, E.P.: Path planning for swarms by combining probabilistic roadmaps and potential fields. In: 14th Annual Conference, TAROS 2013. (2013) 417–428

[9] A. Kamphuis, J.H., Overmars, M.: Motion planning for groups of entities. Eurographics (2003)

[10] Jaillet, L., Simeon, T.: A PRM-based motion planner for dynamically changing environments. In: Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS). (2004)

[11] van den Berg, J., Guy, S.J., Lin, M.C., Manocha, D.: Reciprocal n-body collision avoidance. In: Int. Symposium on Robotics Research. (2009)

[12] Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). (1985) 500–505

[13] Cetin, O., Kurnaz, S., Kaynak, O., Temeltas, H.: Potential field-based navigation task for autonomous flight control of unmanned aerial vehicles. Int. J. of Autom. and Cont. **5**(1) (2011) 1–21

[14] Khuswendi, T., Hindersah, H., Adiprawita, W.: Uav path planning using potential field and modified receding horizon a* 3d algorithm. In: Int. Conf. on Electrical Eng. and Informatics (ICEEI). (2011) 1–6

[15] Weijun, S., Rui, M., Chongchong, Y.: A study on soccer robot path planning with fuzzy artificial potential field. In: Int. Conf. on Comput., Cont. and Industrial Eng. (CCIE). Volume 1. (June 2010) 386–390

[16] Ge, S.S., Cui, Y.J.: New potential functions for mobile robot path planning. IEEE Trans. Robot. Automat. **16**(5) (2000) 615–620

[17] Dolgov, D., Thrun, S., Montemerlo, M., Diebel, J.: Path planning for autonomous vehicles in unknown semi-structured environments. Int. J. Robot. Res. **29**(5) (2010) 485–501

[18] Vadakkepat, P., Tan, K.C., Ming-Liang, W.: Evolutionary artificial potential fields and their application in real time robot path planning. In: IEEE Congress on Evolutionary Computation. Volume 1. (2000) 256–263

[19] Song, Q., Liu, L.: Mobile robot path planning based on dynamic fuzzy artificial potential field method. Int. J. of Hybrid Info. Tech. **5**(4) (2012)

[20] Jaillet, L., Cortés, J., Siméon, T.: Sampling-based path planning on configuration-space costmaps. IEEE Trans. Robot. **26**(4) (2010) 635–646

[21] Patil, S., Van Den Berg, J., Curtis, S., Lin, M.C., Manocha, D.: Directing crowd simulations using navigation fields. Trans.on Visualization and Computer Graphics **17**(2) (2011) 244–254

[22] Kapadia, M., Singh, S., Hewlett, W., Reinman, G., Faloutsos, P.: Parallelized egocentric fields for autonomous navigation. The Visual Computer **28**(12) (2012) 1209–1227

[23] Margellos, K., Lygeros, J.: Hamilton-Jacobi formulation for reach-avoid problems with an application to air traffic management. Amer. Cont. Conf. (2010) 3045–3050

[24] Gillula, J.H., Hoffmann, G.M., Haomiao, H., Vitus, M.P., Tomlin, C.J.: Applications of hybrid reachability analysis to robotic aerial vehicles. Int. J. Robot. Res. (2011) 335–354

[25] Mitchell, I., Bayen, A., Tomlin, C.: A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. Trans. on Auto. Cont. (2005) 947–957

[26] Takei, R., Huang, H., Ding, J., Tomlin, C.: Time-optimal multi-stage motion planning with guaranteed collision avoidance via an open-loop game formulation. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). (2012) 323–329

[27] Ding, J., Li, E., Huang, H., Tomlin, C.: Reachability-based synthesis of feedback policies for motion planning under bounded disturbances. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). (2011) 2160–2165

[28] Majumdar, A., Tedrake, R.: Robust online motion planning with regions of finite time invariance. In: Algorithmic Foundations of Robotics. Springer (2013) 543–558

[29] Abate, A., Prandini, M., Lygeros, J., Sastry, S.: Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems. Automatica (2008) 2724–2734

[30] Summers, S., Kamgarpour, M., Lygeros, J., Tomlin, C.: A stochastic reach-avoid problem with random obstacles. In: Proc. Int. Conf. Hybrid Sys.: Comp. and Cont. (HSCC). (2011) 251–260

[31] Kamgarpour, M., Ding, J., Summers, S., Abate, A., Lygeros, J., Tomlin, C.: Discrete time stochastic hybrid dynamical games: Verification and controller synthesis. In: IEEE Conf. on Decision and Cont. (2011) 6122–6127

[32] Bertsekas, D.P.: Dynamic Programming and Optimal Control. Athena Sci. (2005)

[33] LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. In: Proc. IEEE Int. Conf. Robot. Autom. (ICRA). (1999) 473–479

[34] van den Berg, J., Guy, S.J., Snape, J., Lin, M.C., Manocha, D.: Rvo2 library: Reciprocal collision avoidance for real-time multi-agent simulation `http://gamma.cs.unc.edu/RVO2/`.

[35] Lam, C.P., Chou, C.T., Chiang, K.H., Fu, L.C.: Human-centered robot navigation towards a harmoniously human–robot coexisting environment. IEEE Trans. Robot. **27**(1) (2011) 99–112