IEEE International Conference on Robotics and Automation (ICRA'13), May, 2013

Path Planning with Uncertainty: Voronoi Uncertainty Fields

Kyel Ok, Sameer Ansari, Billy Gallagher, William Sica, Frank Dellaert, and Mike Stilman

***Abstract*— In this paper, a two-level path planning algorithm that deals with map uncertainty is proposed. The higher level planner uses modified generalized Voronoi diagrams to guarantee finding a connected path from the start to the goal if a collision-free path exists. The lower level planner considers uncertainty of the observed obstacles in the environment and assigns repulsive forces based on their distance to the robot and their positional uncertainty. The attractive forces from the Voronoi nodes and the repulsive forces from the uncertainty-biased potential fields form a hybrid planner we call *Voronoi Uncertainty Fields* (VUF). The proposed planner has two strong properties: (1) bias against uncertain obstacles, and (2) completeness. We analytically prove the properties and run simulations to validate our method in a forest-like environment.**

I. Introduction

There exist numerous applications that could benefit from a mobile robot autonomously navigating through a dense environment such as a forest. Wildlife observation, search and rescue of stranded hikers, environment monitoring, and forest fire surveillance are some examples. With these applications in mind, we propose a hierarchical planner for autonomous navigation and exploration in unknown environments and simulate a UAV navigating through a forest-like environment.

Autonomous navigation in an environment with densely distributed obstacles can be achieved using various planning algorithms. However, many of these algorithms assume an accurate prior knowledge of the environment and are often inadequate by themselves for navigating in an unknown environment. State-of-the-art Simultaneous Localization and Mapping (SLAM) techniques attempt to overcome this lack of prior knowledge by gradually building an estimated map of the surrounding environment [1]. However, noise in sensor readings introduce growing uncertainty into these maps.

Typical planning algorithms neglect to address this problem of uncertainty in the information provided. Many works on planning algorithms focus on achieving optimality and reducing computational complexity and often overlook the uncertainty of the information being used. Similarly, typical SLAM algorithms solve localization and mapping problems and do not consider the later usage of the acquired information. Our algorithm solves this lack of connection between SLAM and planning by using the uncertainty of the mapped obstacles in the planning stage to minimize the chance of collision when navigating. This idea belongs to the family of algorithms that combines SLAM and planning techniques together and is commonly denoted as Simultaneous Planning Localization and Mapping (SPLAM).

[†]This work was supported in part by NSF grant IIS-1017076.

The authors are with the Center for Robotics and Intelligent Machines, Georgia Tech, Atlanta, GA, USA. {kyelok, elucidation, bgallagher, dellaert, mstilman}@gatech.edu

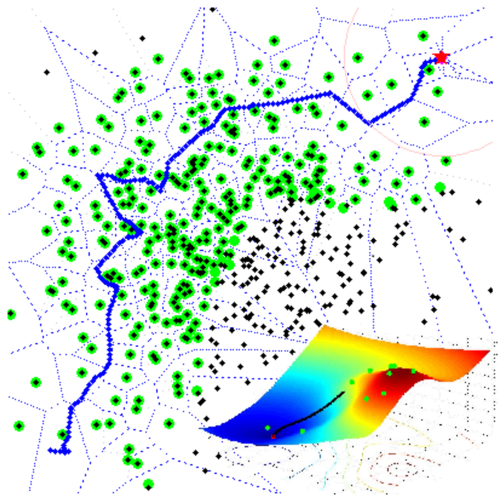

Fig. 1. A simulation of 500 obstacles showing the goal (red star), Voronoi decomposition (thin blue lines), traveled path (thick blue lines), estimated obstacle positions (green circles), and local potential field (bottom-right) for one of the iterations. As illustrated, Voronoi decompositions and potential fields can be combined smoothly with no overhead computations.

Previous works in SPLAM have been geared toward manipulating the planning algorithms to aid the map building and localization processes in SLAM techniques. For example, [2] uses frontier points as local destinations to minimize the system uncertainty and increase the exploration efficiency. Furthermore, [3] presents belief roadmaps (BRM) to demonstrate reducing localization uncertainty by selecting paths with more observable landmarks. Similarly, many works [4], [5], [6] plan trajectories that result in minimal uncertainty in its estimates and the maximal coverage area.

Only few works consider the opposite: navigating more cautiously in regions with high uncertainty instead of attempting to lower the uncertainty of these regions. [7] uses sensor, localization, and mapping uncertainty in SLAM algorithms to bias branch extensions in Rapidly-Exploring Random Trees (RRTs) [8] to nodes with potentially lower localization uncertainty. However, this method fundamentally bases on RRTs and inherits the sampling-based planner's sub-optimality and probabilistic completeness. Our algorithm improves on current works by achieving completeness.

We propose a hierarchical planner, *Voronoi Uncertainty Fields* (VUF), that guarantees completeness while directly accounting for the uncertainty of the observed obstacles. VUF plans a locally best path in terms of shortness and certainty of no-collision, while navigating to a global goal. We discuss the details of our algorithm, shown in Figure 1, in the next section then present simulation results along with a proof of completeness for the planner. Further discussions will follow to analyze and denote areas of improvement.

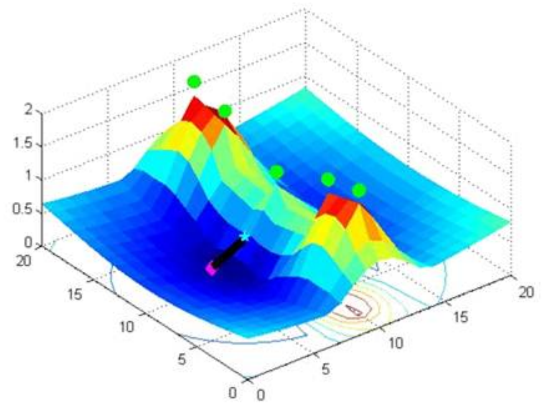

Fig. 2. Gradient descent method trapped by a local minimum, preventing from hitting the wall of obstacles (green spheres). The global planner will pull the robot (magenta) out of this state around the wall if a path exists.

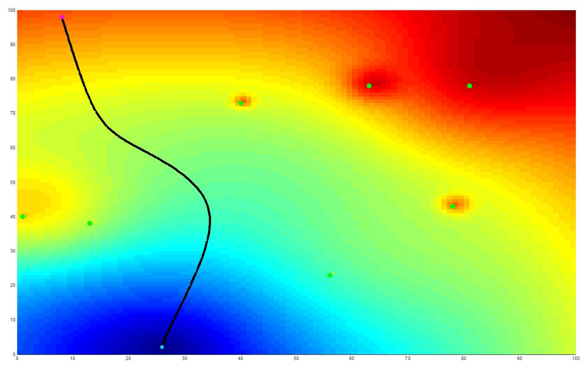

Fig. 3. Potential field contour with densely scattered tree-like obstacles (green circles). The locally best path for the robot (magenta) at the top-left corner away from the obstacles to the local goal (cyan) at the bottom is the gradient descent in the field, as traced in black.

II. Voronoi Uncertainty Fields

Voronoi Uncertainty Fields is a hierarchical planner that has a top-level planner that forms local way-points using Voronoi vertices and a bottom-level planner that locally refines the actual path using uncertainty-biased potential fields.

The top-level global planner uses a modified version of generalized Voronoi diagrams (GVDs) [9] to form a graph of the collision-free space. By traversing through the nodes in the graph, if there is a free path from the source to the goal, it is guaranteed that a path also exists in the graph. Relying on this property, the global planner creates and updates a list of Voronoi nodes that forms the shortest path to the goal. Then, the bottom-level local planner, which fundamentally bases on potential fields [10], chooses the closest node in the list and uses it as an attractor, or a local way-point. Based on the local way-point and the uncertainty-biased repulsive forces exerted by the nearby obstacles, a sum of forces can be calculated for the surrounding local area. This resultant force is then turned into control inputs for the robot.

Although, both Voronoi diagrams and potential fields are well-studied algorithms in the field of Robotics, our method of hierarchically combining and manipulating the algorithms leads to strong properties such as completeness, fast run-time, and overcoming the local minima problem in potential fields, as illustrated in Figure 2. We exploit the division of levels to allow more computationally expensive Voronoi decomposition on the global map to run at a lower frequency while the cheap local planner runs at a much higher frequency to refine the current path and react to immediate threats from nearby obstacles.

There are other planners that exploit the idea of combining multiple planners to compensate for the weakness in each. [11] uses centroid of Voronoi regions as attractors while obstacles and boundary exert short range repulsive forces. Some other planners for mobile robots [12], [13] also use Voronoi diagrams and potential fields together to overcome the shortcomings in each planner. Our work builds on this family of hybrid planners by accounting for the uncertainties in the map, and further coupling with a SLAM system.

A. SLAM

For the localization and mapping of the robot, we assume a generic SLAM system is implemented. The vehicle motion is defined as the following:

$$X_{t+1} = f\left(X_t, u_t\right) + \varepsilon \tag{1}$$

where the robot state X at time $t+1$ is only dependent on the current state X_t, and the control input u_t with zero mean uncorrelated Gaussian noise $\varepsilon \sim N(0, Q_t)$. Similarly, the measurement is defined as

$$Z = h\left(X_t, L_t\right) + \eta \tag{2}$$

where a measurement is a function of the robot state X, and the landmark state L with some Gaussian noise η.

Following a state update using any of the well-known SLAM algorithms such as EKF-SLAM [14], we are interested in the current uncertainty of the landmarks to bias against uncertain obstacles in the planning stage after. This uncertainty in the landmarks is computed by taking the marginal probability of each landmark as follows:

$$P(L_k^t) = \int_v P(X_0) \prod_{i=1}^{t} (X_i \mid X_{i-1}) \prod_{j=1}^{m} P(L_j) P(Z_{ij} \mid X_i, L_j)\, dv \tag{3}$$

where $v = X_{1..t}, L_{1..t} \setminus L_k$.

The marginal probability of the landmarks is approximated by a Gaussian distribution $P(L_j^t) \approx N(\bar{L}_j^t, \Sigma_j^t)$ and the covariance of this distribution is discriminatively used in the local planning stage to bias against uncertain obstacles.

B. Local Planner

After calculating the uncertainty of the obstacles, the local planner takes in a list of the currently visible obstacles and attempts to find a path from the current position to the local goal generated by the global planner. It does this by calculating a potential field and using gradient descent to form the path, as illustrated in Figure 3.

The potential field, p, is comprised of three components as given by Equation 4: 1) the linear distance from the local goal, d_{lg}, 2) the linear distance from an obstacle, d_i 3) the value of a Gaussian distribution, p_i, based on its uncertainty, σ_i. Component 1 ensures that the potential field will push the robot towards its intended goal. Component 2 ensures that the robot won't get too close to any obstacles, and component 3, given by Equation 5, ensures that uncertain obstacles will push the robot further away than well localized ones. Each of these components is empirically weighted by k_j to form a smooth and consistent continuous function. Sensitivity of these weights do not affect the system significantly, as long as each component is in reasonable scale to others.

$$p = k_1 d_{lg} + k_2 p_i + k_3 d_i \tag{4}$$

$$p_i = \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{d_i^2}{2\sigma_i^2}} \tag{5}$$

Given the potential field function, gradient descent methods find the minimum of the function by gradually stepping down the steepest slope iteratively. This is done by estimating the gradient, ∇f, of the function at each step, and taking a step in the $-\frac{\nabla f}{|\nabla f|}$ direction with a step size based on the magnitude of the gradient. This continues until the gradient goes to zero, the step size gets tiny, or the robot reaches its goal.

This method, however, can not traverse uphill and will get trapped by local minima of the potential function. Typically, this is considered problematic, since the algorithm is unable to find a path to the goal. In our algorithm, we overcome the local minima problem by fusing with the global planner. As shown in Figure 2, when the robot is stuck in a local minimum, we simply remove the current local goal from the global planner's graph, and re-find a new shortest path. Accounting for the sensing uncertainty, we do not immediately do this but wait for sufficient measurements to be made. If the robot is detected to be stuck at time s, then the uncertainty in the pose X_s can be computed by marginializing

$$P(X_s) = \int_v P(X_0) \prod_{i=1}^{s} (X_i \mid X_{i-1}) \prod_{j=1}^{m} P(L_j) P(Z_{ij} \mid X_i, L_j)\, dv \tag{6}$$

where $v = X_{1..s-1}, L_{1..t}$. Then, as the robot stays stationary, for a future time k s.t. $k > s$, Equation 3 reduces down to

$$P(L_k \mid Z_k^{1..m}) \approx P(X_s)\, P(L_k) \prod_{i=1}^{m} P(Z_k^i \mid L_k). \tag{7}$$

This marginal probability of obstacle L_k results in more measurement likelihood $P(Z_k^i \mid L_k)$ being multiplied as new measurements are made. Since these are approximately Gaussian, by the well-known property, the resulting Gaussian has a smaller covariance, i.e. over time the uncertainty of the obstacles decrease. Thus, we use a heuristic to re-plan only when the uncertainty of the nearby obstacles drops below a threshold. Note that this approach requires the observation matrix to be full rank, and this limitation is further discussed in a later section.

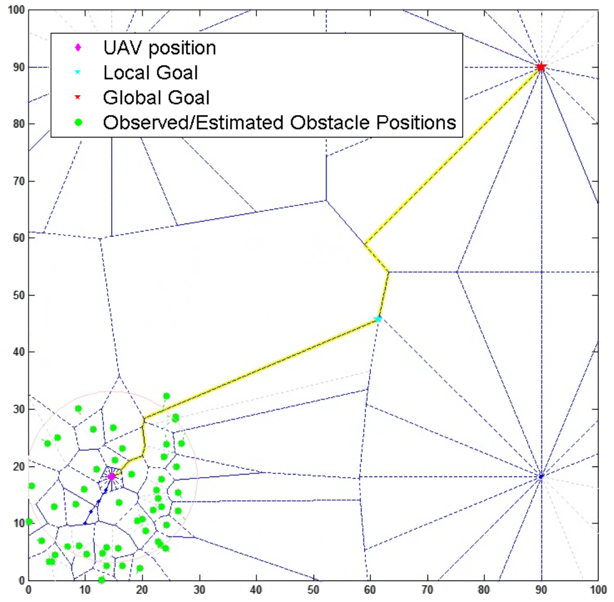

Fig. 4. Global Planner with robot position (purple diamond), global goal (red star), Voronoi edges (blue dotted lines), observed trees (green circles), and the shortest path (yellow solid line) are shown. The polygons added to far-distanced points and the goal, shown at each corner, ensure that Voronoi edges exist for entry into and out of the graph, and in no-obstacle scenarios.

C. Global Planner

The global planner uses generalized Voronoi diagrams[1] to divide the space into Voronoi regions. Given a set S of n obstacles on the ground plane, the dominance of obstacle p over q, where $p, q \in S$, is defined as

$$dom(p, q) = \{x \in R^2 \mid \delta(x, p) \leq \delta(x, q)\} \tag{8}$$

where δ is the euclidean distance function [15]. This dominance is used to decompose the ground plane into regions where each region is dominated by an obstacle. The graph formed by the boundaries of these Voronoi regions is searched by the global planner to generate a collision-free path from the current position to the goal.

However, in order to guarantee completeness and account for cases where Voronoi decomposition fails, several changes are made to the original Voronoi graph. First, it is known that if a collision-free path exists from the start to the goal, then there exists a collision-free path which traverses only the generated GVD graph except for the stages entering and exiting the graph [16]. We address the problem of entering and exiting the Voronoi graph by always including regular polygon of Voronoi sites around the robot and the goal to force Voronoi edges connected to them in the Voronoi graph. As shown in Figure 4, the edges formed by the polygons around the goal create multiple connections to the Voronoi graph in various directions. Also, two far distanced sites are added in similar fashion to deal with the special case of Voronoi decomposition failing with zero obstacles in the map [15]. The Voronoi space is then further processed to prune incorrect edges. First the Voronoi edges that are too close to the obstacles are removed from the graph to account for

[1]If all obstacles are point-like, regular Voronoi diagrams is used instead

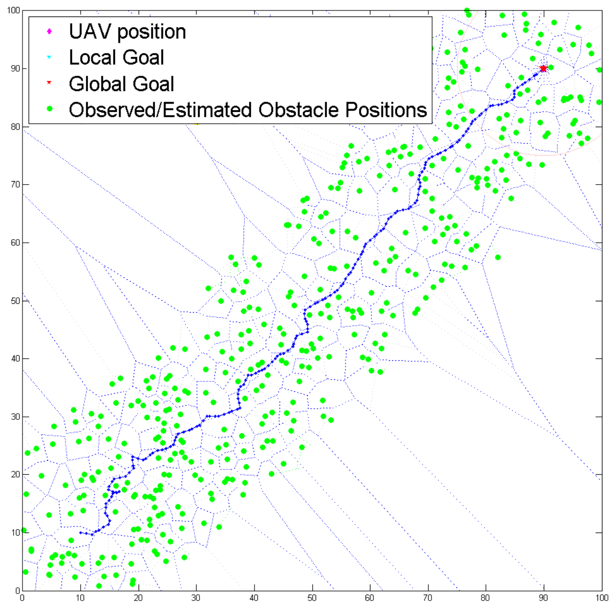

Fig. 5. A simulation of 1,000 obstacles showing the Voronoi decomposition (thin blue), traveled path (thick blue), and estimated obstacle positions (green). Note that the actual path does not exactly follow the Voronoi edges but uses them as local way-points.

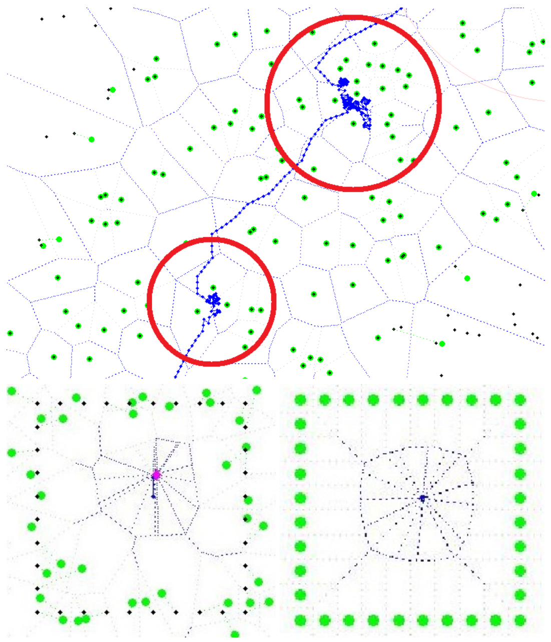

Fig. 6. Simulations of special cases are shown. Top: robot breaking out of two local minima, circled in red. Bottom-left: a caged environment with no solution to the goal. Bottom-right: our complete planner returned no solution after trying all directions.

the robot size and adds some safety margins to avoid tight passages. Each edge v_i that does not satisfy the criteria

$$\delta(v_i, L_k) < D + \tau \tag{9}$$

are removed for all obstacles $L_{1..j}$ where δ is the euclidean distance function, D the robot diameter and τ safety margin. Then, redundant vertices that have no edges connected to them or do not connect to the goal are removed to reduce the graph size. The post-processed graph is searched using A* to find the shortest chain of Voronoi edges from start to goal, and are are stored as a chain of Voronoi nodes between these edges. These Voronoi nodes are later accessed by the the local planner to use as the local goal.

D. Complete Cycle

A complete cycle involves SLAM, local planner, global planner, and a controller all working in a hierarchical order. In a typical cycle, the SLAM algorithm would first localize the robot and measure the nearby obstacles to compute the Gaussian-approximated marginal probability. Based on the created map and the estimated robot position, the global planner would find the Voronoi nodes that form the shortest path in its graph space. The local planner would then segment out a small region with nearby obstacles and the local goal on the global path. Using the Gaussian uncertainties and the information about the local area, the local planner would generate a direction most desirable in terms of path shortness and certainty of obstacle positions. A controller would then actuate the robot towards desired directions, until either the local goal is reached or the global planner updates the local goal. This cycle repeats until the robot is within a certain distance to the global goal. We simulate this cycle in MATLAB, using several different scenarios.

III. Experiments

We have simulated Voronoi Uncertainty Fields in MATLAB, in random environments with tree-like obstacles. The start and goal positions were arbitrary set at a far distance from each other and a bounding box was made to constrain the environment. In order to simplify the full navigation problem to fix the scope on the planner, we abstract out the measurement, SLAM and controls problems. For the measurement, the observation matrix is assumed to be full-rank where the information about the obstacles nearby are obtained each iteration with uncertainty decreasing with more measurements. At each iteration, we consider only the uncertainty in the landmarks in the current local frame, and assume that a graph of such local frames can be optimized as done in relative SLAM works such as [17]. Finally, perfect actuation is assumed to abstract out the controls problem.

We simulated different scenarios with varying numbers and positions of obstacles. The obstacle number ranged from 0 to 10,000 obstacles and the positions were either random placements or normal distributions. Some tests for special cases were also performed to verify our planner's ability to break out of local minima, and terminate if no solution exists.

1) 1,000 Trees: In Figure 5, a 1,000 obstacle scenario is shown with only the information known to the planner. Other than the surrounding area en route to the goal, the rest of the area out of the sensor range is not revealed.

2) Escaping Local Minima: Another scenario simulated is to verify the planner's ability to escape local minima. Shown in Figure 6, there are two local minima that temporarily block the robot. The global planner, which keeps track of last known positions, is able to choose a new local goal that navigates the robot out of the local minima.

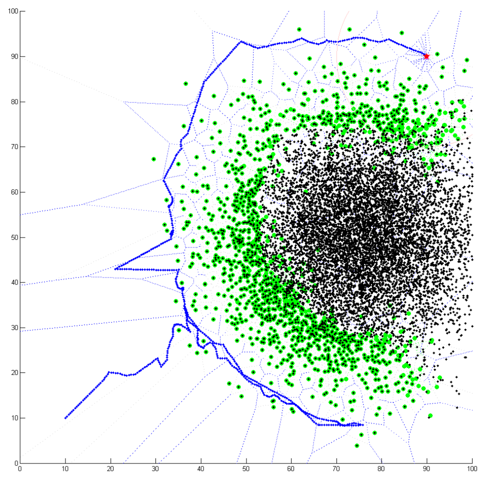

Fig. 7. Stress testing the planner with 10,000 obstacles in the environment. The planner takes a long detour but eventually arrives at the goal.

3) No Solutions: To verify the completeness of our planner, a few scenarios with no solutions were also simulated. Shown in Figure 6, when the planner was run in an environment consisting of a robot boxed in a square of obstacles with the goal outside the box, the planner continued to search for a path until uncertainty of the obstacles dropped below a threshold, where the planner terminated with no solution.

4) 10,000 Trees Stress Test: A large scenario with 10,000 normally distributed obstacles was used to test the planner's ability to utilize the global planner's ability to back-track and find new paths when detecting dead ends. As shown in Figure 7, obstacles were normally distributed at an offset from the center of the map, and the robot was started in the bottom-left corner with the goal in the top-right corner of the field. As seen in the path, the robot initially travels directly towards the goal, but upon encountering obstacles it navigates to the right counter-clockwise searching for a path. At a certain point at the bottom of the mass of obstacles, it determines that the cost of traveling any further counter-clockwise is more than it is to back-track and navigate clockwise, which it then does and successfully reaches the goal.

In our simulations, we have demonstrated that (1) the robot is able to back-track and reach the goal, (2) escape local minima, and (3) return with no solution if there is no path. In the following section, we analyze and discuss the observed properties and prove that our planner is complete.

IV. Analysis

A. Completeness

We define our workspace W as a bounded 2-dimensional space (i.e. ground plane for a ground robot). The workspace can be Voronoi decomposed into a configuration space V of Voronoi regions, $V = [v_1, ...v_n]$, using Equation 8. If all sites, or obstacles in the workspace, are not collinear, each Voronoi region v_i has a connected boundary and the Voronoi diagram is connected, as proven in Theorem 7.4 [18].

Then, traversability from one Voronoi region v_p to a neighboring Voronoi region v_q, across one segment of its connected boundary (commonly termed Voronoi edge) can be calculated using Equation 9. Our configuration space V of finite number of regions $v_{1..n}$ can then be represented as a graph where the nodes are Voronoi edges of these connected regions and a graph edge between two nodes is traversability between the two regions. It is well known that A* search on such graph for a path from one node v_p to to another v_q is complete [19]. With pre-processing to remove collinearity, our global Voronoi planner that uses this search to traverse from current Voronoi region to the goal region along the boundary edges is then complete.

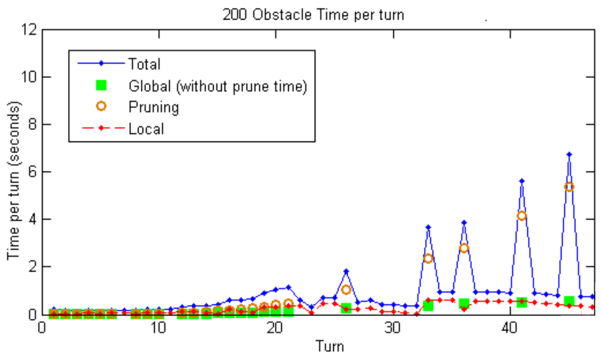

Fig. 8. Timing graph showing the time taken for the local planner, the global planner (not accounting for the pruning time), the pruning time, and the total combined time. It can be observed that most overhead comes from the inefficient pruning stage while both planners are near real-time performance without any code optimizations.

B. Optimality

Contrary to completeness, optimality is not guaranteed for the global planner. Since the representation of the world is constantly updating as the robot progresses, it is not possible to have a globally optimal solution before all regions have been explored. Although A* search with admissible euclidean distance heuristic used in the global planner does obtain the shortest path to the goal, it is only the shortest distance in the Voronoi graph, and may not be the shortest path in the region.

C. Computation Time

Timing tests were done in a uniform environment to explore the possibilities of real-time operation of the planner. Table I shows the time taken on average for a single iteration of the global and the local planner in a map of 200 obstacles with pruning of the Voronoi graph turned on and off.

While the local planner is fast and only uses a local subset of obstacles, the global planner uses every obstacle observed and loses scalability. For example, navigating in the map of 200 obstacles, the local planner only interacted with 34.4 obstacles on average, whereas the global planner interacted with 139 obstacles in the last ieration. Despite the scalability issue, both the global and the local planners are well manageable for real-time operations as evidenced in Figure 8 as long as the pruning of the Voronoi graph is turned off. Efficient implementation of the pruning stage remains as an area of improvement for real-time operations.

TABLE I

TIME TAKEN FOR A SINGLE ITERATION AT THE BEGINNING AND THE END OF THE OPERATION FOR BOTH LOCAL AND GLOBAL PLANNERS.

200 obstacles (without pruning)			
Iteration	Total (s)	Global Plan (s)	Local Plan (s)
First	0.528	0.0416	0.0822
Last	1.026	0.590	0.0572
200 obstacles (with pruning)			
Iteration	Total (s)	Global Plan (s)	Local Plan (s)
First	0.333	0.051	0.022
Last	12.233	10.477	1.025

V. DISCUSSION AND FUTURE WORK

Voronoi Uncertainty Fields (VUF) is a new deterministic, complete collision-free algorithm for navigation in uncertain environments. Results in Figure 7 show that the global map generated during the planner's operation can be used to re-plan when a dead end is encountered. In Figure 6, it is evidenced that the global Voronoi planner can move the robot out of local minima and correctly terminate when a collision-free path to the goal does not exist.

Although the simulations demonstrate our planner's capabilities in ideal situations, extending the planner to use on an actual mobile robot requires solving the associated observation, map/pose optimization, and control problems that were abstracted out in the simulation of the proposed planner. For example, current observation model assumes a hypothetical sensor with a full-rank observation matrix and may pose a problem in realistic implementations. However, this assumption is only used to reject false-positive local minima when stuck, and may not be needed with a different strategy to deal with local minima, i.e. breaking out immediately when stuck and exploring the nearby area or re-visiting only after no other solutions exist. In addition, the limitation in the system that the measurement noise must be Gaussian may need to be overcome for realistic low-cost sensors.

While the controls problem is platform specific and the current algorithm does not impose any constraints, it would be desirable to factor in the velocity of the vehicle in the planner to allow only the paths that can be executed.

It is also possible to extend current algorithm in 2-dimensional space to higher dimensions by replacing the global planner with a high-dimensional planner such as the RRTs [8]. As long as the replacement global planner can find a feasible path to the goal and iteratively provide a local goal to the local planner, i.e. providing the nearest tree node on a viable path to the goal when replacing with the RRTs, the local planner would still take account of the uncertainty in the obstacles and refine the local path. Moreover, such choice of global planner could also extend uncertainty to be handled in the global level as well by using techniques such as biasing node sampling away from uncertain regions [7]. However, a drawback of replacing the global planner is that the overall system may lose the the properties of the Voronoi planner, such as completeness if the replacement planner does not have such properties.

Looking at the timing results of the simulations in Table I, our planner could potentially operate in real-time with a large number of obstacles, once some improvements in the pruning stage is made and the scalability issue is dealt with. To gain scalability, a solution that allows the global planner to limit the number of obstacles it processes each step while preserving the capability to backtrack upon encountering a dead-end would be desirable. One possible solution would be to further divide the hierarchical approach and have the global planner use a subset of obstacles, and only use the full global map when no solution exists in the local path.

Although some areas of improvement remain as future work, we have proposed a novel planner, Voronoi Uncertainty Fields, that deals with map uncertainties in a deterministic and complete way.

REFERENCES

[1] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[2] T. Tao, Y. Huang, F. Sun, and T. Wang, "Motion planning for slam based on frontier exploration," in *Mechatronics and Automation, 2007. ICMA 2007. International Conference on*. IEEE, 2007, pp. 2120–2125.

[3] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in belief space by factoring the covariance," *The International Journal of Robotics Research*, vol. 28, no. 11-12, p. 1448, 2009.

[4] T. Kollar and N. Roy, "Trajectory optimization using reinforcement learning for map exploration," *The International Journal of Robotics Research*, vol. 27, no. 2, p. 175, 2008.

[5] R. Sim and N. Roy, "Global a-optimal robot exploration in slam," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE, 2005, pp. 661–666.

[6] C. Leung, S. Huang, and G. Dissanayake, "Active slam in structured environments," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 1898–1903.

[7] Y. Huang and K. Gupta, "Rrt-slam for motion planning with motion and map uncertainty for robot exploration," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, 2008, pp. 1077–1082.

[8] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *In*, no. 98-11, 1998.

[9] D. Lee and R. Drysdale III, "Generalization of voronoi diagrams in the plane," *SIAM Journal on Computing*, vol. 10, p. 73, 1981.

[10] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, vol. 2. IEEE, 1985, pp. 500–505.

[11] A. Renzaglia, A. Martinelli, *et al.*, "Distributed coverage control for a multi-robot team in a non-convex environment," 2009.

[12] E. Masehian and M. Amin-Naseri, "A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning," *Journal of Robotic Systems*, vol. 21, no. 6, pp. 275–300, 2004.

[13] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous driving in unknown environments," in *Experimental Robotics*. Springer, 2009, pp. 55–64.

[14] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping (slam): Part i the essential algorithms," *IEEE ROBOTICS AND AUTOMATION MAGAZINE*, vol. 2, p. 2006, 2006.

[15] F. Aurenhammer, "Voronoi diagrams&mdash;a survey of a fundamental geometric data structure," *ACM Comput. Surv.*, vol. 23, no. 3, pp. 345–405, Sept. 1991. [Online]. Available: http://doi.acm.org/10.1145/116873.116880

[16] O. Takahashi and R. Schilling, "Motion planning in a plane using generalized voronoi diagrams," *Robotics and Automation, IEEE Transactions on*, vol. 5, no. 2, pp. 143–150, 1989.

[17] E. Eade and T. Drummond, "Monocular slam as a graph of coalesced observations," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, oct. 2007, pp. 1 –8.

[18] M. De Berg, O. Cheong, and M. Van Kreveld, *Computational geometry: algorithms and applications*. Springer-Verlag New York Inc, 2008.

[19] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *Systems Science and Cybernetics, IEEE Transactions on*, vol. 4, no. 2, pp. 100–107, 1968.