The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Prioritized Sensor Detection via Dynamic Voronoi-Based Navigation

R. Andres Cortez, Rafael Fierro, and John E. Wood

Abstract— **This paper presents a decentralized coordination algorithm that allows a team of sensor-enabled robots to navigate a region containing non-convex obstacles and take measurements within the region that contain the highest probability of having "good" information first. This approach is motivated by scenarios where prior knowledge of the search space is known or when time constraints are present that limit the amount of area that can be searched by a robot team. Practical applications include search and rescue, target detection, and hazardous contaminations. Our cooperative control algorithm combines Voronoi partitioning, a global optimization technique, and a modified navigation function to prioritize sensor detection. The issues we address such as non-convex obstacles as well as global search are not extensively addressed in the current literature. Simulation results of the control algorithm are given and validate the prioritized sensing behavior as well as the collision avoidance property.**

I. Introduction

Robotic motion planning is a well-addressed issue in autonomous systems, [1]. However a growing number of applications such as spatial distribution mapping, dynamic sensing coverage, and dynamic target detection, have motivated navigation and control algorithms for teams of goal-oriented mobile sensor networks. When considering control and coordination algorithms of reconfigurable sensor networks one must join the coordination/navigation of the robots with the sensing cost or desired configurations of the sensor network. In problems involving reconfigurable sensor networks, a primary goal is to reconfigure the sensor network in such a way that the time taken to reconfigure is minimized or the sensing coverage is maximized. This has useful applications in target detection and surveillance as well as spatial distribution mapping, among many others.

In general, approaches to reconfigurable sensor networks use gradient type algorithms to reconfigure the network for sensing optimality [2]. Using a gradient type approach has been shown to be successful when the underlying density function is static and the environment is free of obstacles. These local gradient type approaches which address sensing coverage problems cannot however, address certain types of reconfigurable sensor network problems. One very interesting problem, which we consider here, is that of biasing certain regions of the area-of-interest to be searched because of prior knowledge or because an underlying time constraint that prohibits an exhaustive search of the area, i.e., emergency situations, search and rescue. Gradient type approaches may fail to send the robots to places that have the highest likelihood of containing the most useful information in the time allotted because of the local nature of the gradient technique. Also the probability of certain regions containing useful information will be changing as the robots search the area, which makes the underlying density function in this scenario dynamic. The problem here becomes, how to coordinate the sensor-enabled robot team in such a way that areas within the region that have the most probability of containing "good" information are searched first. The label "good" here may mean a target of interest, hazardous material, a group of people to be rescued, etc., depending on the particular application the robot team is tasked with. Because of prior knowledge, or due to time constraints, the robots should search the most likely areas of finding "good" information first and continue until enough information is gathered or the time constraint is met. The scenario stated here is quite different from other sensor network problems because the reconfigurable sensor network is not trying to achieve optimal sensing coverage over the area-of-interest nor exhaustively search the space, rather the robot team is trying to search the most probable places of containing "good" information first because the search is biased by prior knowledge or because of a time constraint may greatly limit the robot team's ability to search the entire area of interest.

Some motivating and practical applications include search and rescue operations [3], target detection [4], and hazardous contaminations [5] to name a few. In these scenarios, regions within a given area that are most likely to contain humans, enemy targets, or hazardous material, should be searched first while regions with less probability of containing these features are searched later.

The contribution of this paper is a decentralized coordination algorithm for a team of sensor-enabled robots (reconfigurable sensor network) that allows a team to navigate from an initial configuration within a region containing general shaped obstacles, to areas within the region with the highest probability of containing "good" information first, while less probable areas are searched later. Our solution to this problem is based upon Voronoi partitions, a global optimization technique [6], and a modified navigation function [7]. Voronoi partitioning allocates different regions within the area of interest among the robots. These partitions utilize robots to search different areas in parallel, which speeds up the detection process. The global optimization algorithm then computes goal points within the Voronoi partitions and

R.A. Cortez and J. Wood are with the Department of Mechanical Engineering, University of New Mexico, Albuquerque, NM 87131-0001, USA {acortez, wood}@unm.edu

R. Fierro is with the MARHES Lab, Electrical & Computer Engineering Department, University of New Mexico, Albuquerque, NM 87131-0001, USA rfierro@ece.unm.edu

the modified navigation function guarantees collision free motion to the desired goal points.

II. Problem Formulation

Consider a team of n kinematic agents each equipped with a sensor having a sensing radius R, each tasked with starting from some initial configuration and navigating to visit regions within the area which contain the highest probability of detection first, while avoiding collisions with other robots and obstacles within the environment. Let the probability of detection (POD) map reflect the probability of detecting a target over the area to be searched. Define β, to be a parameter that reflects the reduction in the probability of detection map for points inside each robots sensing radius. Consider the area-of-interest $Q \subset \mathbb{R}^2$, with boundary ∂Q, to be a simple convex polygon. Let us denote the POD map as $M(q)$, where $q \in Q$. Q is populated with N_o fixed, general shaped polygonal obstacles $C_{obs} = \{O_1, \ldots, O_{N_o}\} \subset Q$. Define $Q_{free} = Q \setminus C_{obs}$, as the set of points free of obstacles. Here we assume that C_{obs} is known *a priori* by all the robot team members. Each agent p_i is assumed to be holonomic with dynamics:

$$\dot{p_i} = u_i,\ i = 1, \cdots, n \tag{1}$$

where $u_i \in \mathbb{R}^2$ is the control input of agent i.

A. Voronoi Partitions

We partition the area between the available mobile robotic platforms using Voronoi partitions in which the centroid of each Voronoi cell is taken to be the position of a single mobile robot. Thus, a certain region within this area (namely the corresponding Voronoi cell) is allocated to each robot for searching. This is performed on an iterative basis, so the Voronoi partitions are dynamic in nature.

Using Voronoi partitions, the area to be mapped can be broken up dynamically among the robot team members based on their current locations. Also by construction, Voronoi partitioning can be implemented in a decentralized fashion.

Our area to be searched, Q, is assumed to be a simple convex polygon in $\mathbb{R}^2$ including its interior. Let P be a set of n distinct points $\{p_1, \ldots, p_n\}$ that reside in the interior of Q. Define the *Voronoi Partition* of the convex polygon Q, generated by P to be the set of all points in Q such that all points in the region $V_i(P)$ are closer to p_i than any other point in Q,

$$V_i(P) = \{q \in Q \mid \|q - p_i\| \leq \|q - p_j\|, \forall p_j \in P\}.$$

Let us now define the set $D_i = V_i \cap Q_{free}$. D_i represents the set of all points in V_i that are not occupied by an obstacle. Notice that to define the boundaries of each Voronoi cell, robot i at position p_i only needs to know the boundary of Q, and the positions of its nearest *neighbors*. A *neighbor* of robot i (in the sequel identified with its position p_i) is any other robot p_j, such that the Voronoi cells of p_i and p_j share a common edge.

Dividing up the area to be explored in this way, keeps us from assigning robots to specific regions which may not be implementable in a decentralized fashion [8]. Using dynamic Voronoi partitions each robot can compute its partition with only knowledge of its neighbors' locations. Thus, using Voronoi partitioning facilitates decentralized control designs. Another advantage to using Voronoi partitions is in the case where there is a robot or sensor failure. Because the Voronoi partitions are made dynamically, the team can adjust their Voronoi partition configuration taking into account their new *neighbors* excluding the failed robot. This procedure insures that all regions in the area will be covered by a corresponding robot.

B. Global Optimization

There exists a number of algorithms that can solve the global optimization problem for a general (non-convex) function such as a branch and bound technique, evolutionary algorithms, and randomized algorithm methods [6]. However, because our motivation for this research is focused on a decentralized control algorithm, many of these global optimization techniques are not applicable because of computation power needed or because of the on-line nature of the control algorithm. Also another key point to note is that the probability of detection map is being updated at each new iteration. As regions are searched by the robots, the POD map is decreased in those regions within the sensing radius. This could possibly make the POD map vary greatly from one point that has been searched to an adjacent point that has not been searched by the robot. This limits our ability to make any claims as to certain properties the map (function to be optimized) will exhibit, e.g., Lipschitz continuity or if one can compute lower bounds for feasible regions, which are needed for branch and bound techniques. Also, evolutionary algorithms cannot strictly guarantee convergence to a global optimum. As a result of the limitations imposed by the decentralized, on-line design of our algorithm, we propose the use of a randomized algorithm [6] to compute an approximate solution to the global optimization problem. A randomized algorithm does not make any assumptions on the function to be optimized or the feasible set of possible solutions and can be efficiently computed. For our scenario, a function evaluation in practice boils down to looking up values in the "probability of detection" map. Here we will state the general Monte Carlo method used in our approach.

Let the objective function $g(x)$, $x \in A \subset \mathbb{R}^n$ and the set A be measurable where $g^*(x)$ is the global maximum of $g(x)$. To approximate the global maximum $g^*(x)$,

- Generate $X_1, \ldots, X_N$ independent identically distributed (i.i.d.) samples from a p.d.f. $f(x)$ such that $f(x) > 0$.
- Find $Y_k = g(X_k),\ k = 1, \ldots, N$.
- Estimate g^* by $\tilde{g}^* = \max(Y_1, \cdots, Y_N)$.

It is known that in general there does not exist a stopping condition for an achieved accuracy $\tilde{g}^* - g^* < \epsilon$ for the approximate maximum to the global optimization problem [9]. However one can look for stopping conditions to the global optimization problem in statistical terms.

The probability of sampling at least one point inside the region of attraction of the global maximum when taking N sample points from A is given by

$$Pr = 1 - (1 - \alpha_1)^N, \tag{2}$$

where α_1 is the probability of sampling a point in the region of attraction in one trial. By choosing a lower bound on α_1 and a required "accuracy" level Pr, the number of function evaluations N needed to achieve a $\tilde{g}^*$ within the accuracy level can be calculated [9].

C. Navigation Function

A navigation function is an artificial potential function with a unique minimum located at a goal point whose domain of attraction includes the entire domain excluding the points covered by obstacles. The construction of such a navigation function was first shown by Rimon and Koditschek [10], and slight variations have since made their way into the literature [11], [12]. A recent approach to navigation functions has combined robot navigation and communication constraints [13]. One drawback to these particular constructions of navigation functions is that they are difficult to compute in general and also do not easily lend themselves to general shaped obstacles. To overcome this limitation we consider the construction of a navigation function from [7] , which relaxes the requirements on the navigation function, primarily that the gradient of the navigation function need only be piecewise continuous and that the navigation function at the boundaries need not be uniformly maximum. The construction of a navigation function in this way still guarantees one unique minima at the goal point.

Navigation functions involving multiple robots generally have each robot consider all other robots to be moving obstacles [11]. In our approach however, we partition the area among the robot team members, then each robot creates its own navigation function based on its Voronoi partition V_i, the known obstacles in the area C_{obs}, and the goal point calculated from the global optimization algorithm. In this way each robot does not need to consider other robots as obstacles, since by construction the Voronoi partition V_i is unique to robot p_i. Our approach has the advantage of eliminating the need to keep track or "predict" where these "dynamic" obstacles are or will be in the future.

Alternative planning methods for addressing obstacle avoidance are based on cell decomposition approaches. Cell decomposition is a well-known obstacle avoidance method that decomposes the obstacle-free robot configuration space into a finite collection of non-overlapping convex polygons, known as cells, within which a robot path is easily generated. Although it is computationally intensive, its advantage over other robot path-planning approaches, such as roadmap or potential field methods, is that, under proper assumptions, cell decomposition is resolution complete. Exact cell decomposition is guaranteed to find a free path, whenever one exists, and otherwise to return failure. Its disadvantages are that it is computationally intensive in high-dimensional configuration spaces (e.g., robot manipulators), and that it does not typically allow the user to incorporate other motion constraints, such as, nonholonomic dynamics, or sensing/communication constraints. Also, it is not directly applicable to cooperative networks, in which the path of one robot is influenced by that of the other agents in the network.

III. Methodology

In this section we present our multi-vehicle coordination algorithm. The algorithm combines Voronoi partitions, which divide the area to be searched among the robot team members, a Monte Carlo optimization technique, which is used to calculate goal points that correspond to points inside the Voronoi partitions that have the highest probability of containing "good" information, and a modified navigation function that steers the robots from their current positions to their respective goal points while avoiding collisions with the environment.

A. Modified Navigation Function

A navigation function, $f(p, g_p, d)$ which is a function of p, the robots current position, the goal point g_p, and points inside the Voronoi partition not occupied by obstacles $d \in D$, can be created by the following procedure [7].

- Make a graph out of the rectangular mesh of the obstacle grid map, with vertices at the corners of each square and edges along the square edges. Remove vertices and edges that are in the interior of obstacles.
- One of the vertices is chosen as the goal point, determined from the global optimization algorithm.
- Solve the shortest-path problem in the graph. Mark each vertex with the corresponding path length, and let this length be the value of $f(\cdot)$ at the vertex.
- Divide the squares into triangles by drawing a diagonal through the corner with the highest $f(\cdot)$ value.
- In the interior of each resulting triangle, let $f(\cdot)$ be a linear interpolation between $f(\cdot)$ at the three vertices.

Note that the shortest-path problem in the graph can be solved with polynomial time algorithms [14]. Also the shortest-path problem to create this navigation function is solved for a 4-neighborhood grid, i.e., L_1 distance.

B. Control Algorithm

From design, the control algorithm is decentralized and executed in parallel on all robots. The general scheme is outlined below.

1) (*Voronoi region*) For $i = 1, \cdots, n$ determine the Voronoi partition V_i in Q.
2) (*Global optimization*) Apply the general Monte Carlo optimization method over the set D_i to determine an approximate maximum $\tilde{g}^*$ in D_i of $M(q)$.
3) (*Check feasibility*) Determine if $\tilde{g}^*$ is reachable by solving the shortest-path problem from the graph that creates the navigation function. If the point is reachable set the goal point $g_{pi} = \tilde{g_i}^*$. If $\tilde{g_i}^*$ is not reachable then go to Step 2 and determine

$$\tilde{g_i}^* = \max(Y_1, \ldots, Y_{N-1})$$

where we exclude $\tilde{g_i}^*$ that was calculated from the previous optimization.

4) (*Navigation function*) Create a navigation function, $f_i(p_i, q_{pi}, d)$ in D_i with g_{pi} set as the goal point.
5) (*Control actuation*) Let $u_i = \dot{p}_i$ and apply the control

$$u_i = \begin{cases} -k\nabla f_i(\cdot) & \text{if } |u_i| < u_{\max} \\ -\frac{k\nabla f_i(\cdot)}{\|k\nabla f_i(\cdot)\|} \cdot u_{\max} & \text{otherwise} \end{cases} \quad (3)$$

where $k = |p_i - g_{pi}|$ the distance of the robots current position p_i to the goal point g_{pi}.
6) (*Local map update*) For all points that are inside the sensing radius R, update the M as,

$$M(p_i) = \beta M(p_i).$$

7) (*Global map update*) Robot i communicates with its neighbors and exchanges its current position. All robots update their local maps with all other robots local maps to create a synchronized global map.
8) (*Termination*) Check if $t \geq T_{search}$ if true, stop. Else goto Step 1. T_{search} is taken to be the time allowed for the search.

In step 3 we check the feasibility of the goal point obtained from the optimization algorithm. This is done because there may exist configurations of obstacles in the Voronoi partition that cause points in V_i to be unreachable by the robots. If the goal point obtained from the optimization algorithm is unreachable, that point is disregarded and the next best point is taken. It is key to note that there will always exist a feasible goal point for all the robots because in the most improbable case the goal point would be the current robot position. Notice that this case does not however cause robots to become "stuck," since at each iteration of the algorithm the Voronoi regions are updated based on the new positions of the robots at the subsequent optimization is done over the new Voronoi partitions. Also notice that in step 7 we assume that all robots can synchronize their local maps to create a global map. This may seem like somewhat of a limitation on the ability of the control algorithm to be implemented in a decentralized fashion, however it has been shown that this can be done using only one-hop neighbors in the communication network [15]. Although partitioning the search space among the robot team members does place a constraint on the optimal solution, it's usefulness is seen in the fact that it keeps multiple robots from visiting adjacent points near the global optimum which may be searched by a single robot as well as insuring collision avoidance among the robots. The partitioning also facilitates a more expansive search in the initial stages of the algorithm.

C. Properties of Control Algorithm

Proposition 1: (Safety) The control algorithm outlined above guarantees collision avoidance with the environment as well as with other robot team members.

Proof: Notice that by construction, for any robot configuration $P = \{p_1, \ldots, p_n\}$ the accompanying Voronoi partitions $V_1, \ldots, V_n$ are disjoint. Also by construction the navigation function, $f_i(\cdot)$ is defined over only $D_i = V_i \cap Q_{free}$ for all n robots. Therefore any robot i starting inside D_i and following the control described in (3) will stay inside D_i. This guarantees no inter-robot collisions will occur. Obstacle avoidance within D_i comes directly from the construction of the navigation function and the control law described in equation (3). ■

It is also key to note that convergence to the goal point is also guaranteed using a navigation function and the control law (negated gradient) in equation (3), [10]. This is true since by construction the navigation function $f_i(\cdot)$ contains only one minimum at the goal point. A well known result from calculus guarantees that following the negative gradient of a function containing a single minimum will converge to the unique minimum.

IV. Simulation Results

To illustrate our coordination algorithm we choose a region consisting of a 30×30 grid with two general shaped obstacles. For this simulation we use four robots with initial positions clustered near the center of the region. In Figure 1 we see the initial probability of detection (POD) map which reflects regions in the area that may contain "good" information. We take the parameter $\beta = 0.2$, which reflects the reduction in the probability of detection map for points that lie within each robots sensing radius. The search time T_{search} is taken to be 75 iterations of the control algorithm. The velocity of the robots are constrained to $u_{\max} = 1$ and each robots sensing radius is set to $R = 1$cell. For the optimization we take $N = 390$ function evaluations based on an accuracy level of 98% and a lower bound $\alpha_1 = 0.01$.

After the time threshold of 35 iterations is met, we see the reduction in the POD map, Figure 1 (bottom figure). We notice that the map has been reduced significantly. Figure 1 (bottom figure) also illustrates why a Monte Carlo type method is useful to compute the global maxima for each Voronoi region. Notice the very "jagged" nature of the map after it has been reduced.

Figure 2 shows the trajectories of the robots for a single iteration of the algorithm. Notice how the robots navigate only inside their corresponding Voronoi partitions and avoid collisions with obstacles in the environment.

Plots of the navigation functions, their contours, and corresponding robot trajectories created for one iteration of the simulation are shown in Figures 3 - 6. Notice how the contours change around the obstacles to insure collision avoidance, also note that there exist only one unique minimum in each Voronoi region. For the calculations of the Voronoi partitions we enlisted the help of the Multi Parametric Toolbox [16].

V. Conclusions

This paper presented a decentralized control algorithm for a team of sensor-enabled robots to navigate within a region containing general shaped obstacles to areas within the region that contain the highest probability of having

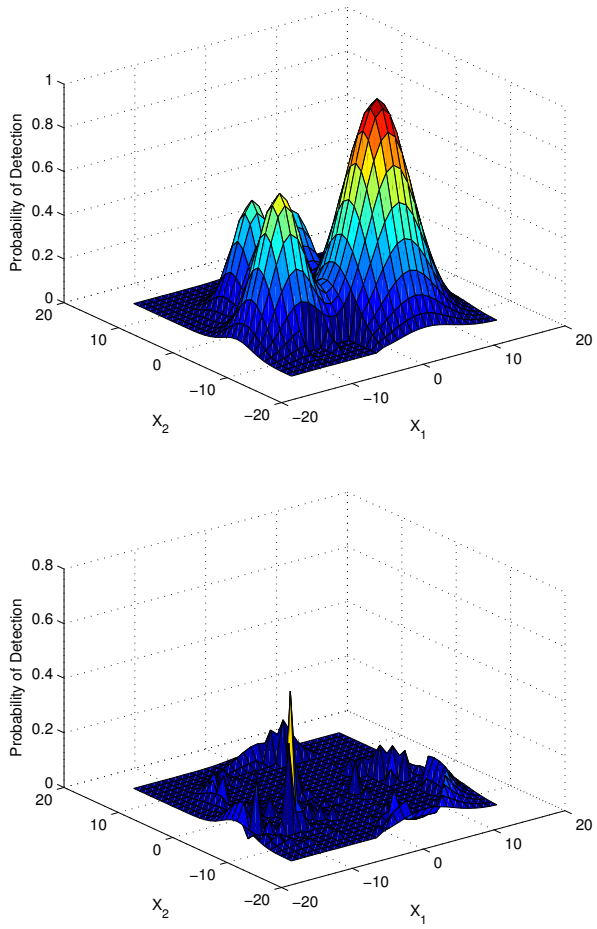

Fig. 1. The top figure shows the probability of detection map $M(q)$ which reflects the likelihood of detecting "good" information in the region. The bottom figure shows the probability of detection map after 35 iterations of the control algorithm. Notice that the probability of detection has been reduced significantly.

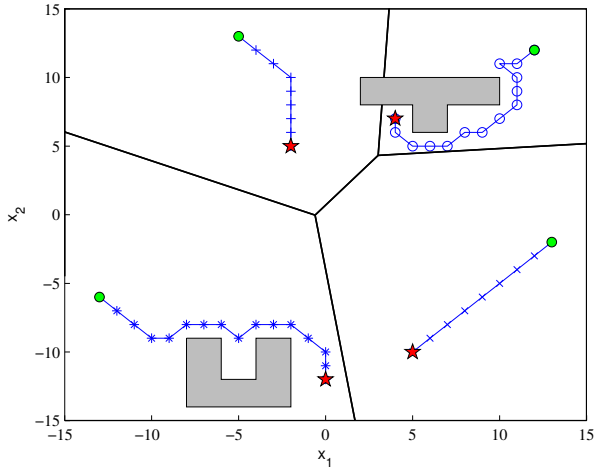

Fig. 2. The robot's trajectories from an initial configurations to the respective goal points for a single iteration of the algorithm. The black lines represent the Voronoi partitioning of the group. The green circle denotes the robots initial positions and the goal points are denoted with a red star. Notice how the robot goes around the obstacle to avoid a collision while not leaving their respective Voronoi partitions.

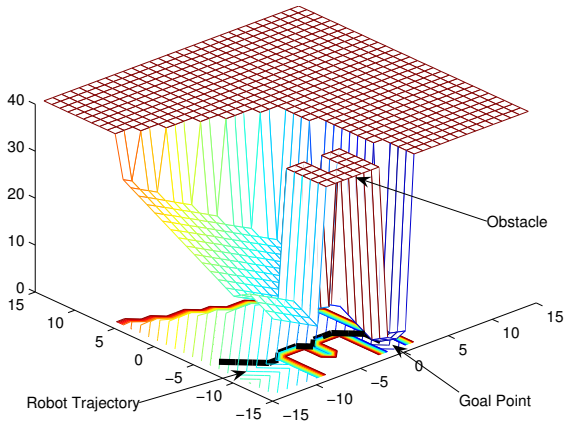

Fig. 3. The plot of the navigation function, it's contours, and the trajectory for the first robot in the lower left hand corner of the region. The entire obstacle in the lower left hand corner intersects the first robot's Voronoi partition.

"good" information first. The control algorithm guarantees collision avoidance with the environment and other robots. Our approach has the advantage of eliminating the need to keep track or "predict" where "dynamic" obstacles (other robots) are or will be in the future in the computation of the navigation function. Another advantage of the control algorithm over other techniques in reconfigurable sensor networks is enabling the robot team to search areas with the highest probability of containing "good" information first, thus prioritizing the search to areas that are believed to have a better chance of containing useful information The problem addressed in this paper also creates a way to prioritize robot searches without having to explicitly task certain robots with a list of goal points to reach. Instead a probability map can be created and then relayed to the robot team who then handles the task assignment. This type of search technique has very practical applications in search and rescue missions, target detection, and hazardous contaminations to name a few. Simulations verified the prioritized sensing behavior as well as the collision avoidance properties.

Future work will entail implementing the decentralized control algorithm on a team of four Pioneer 3-AT robots at the MARHES laboratory located at the University of New Mexico [17]. Other applications of the decentralized control algorithm will look at integrating robot team members with different sensing modalities i.e., heterogenous robot teams, as well as different sensing footprints similar to [18].

ACKNOWLEDGMENT

The work was supported by DOE URPR (University Research Program in Robotics) grant DE-FG52-04NA25590

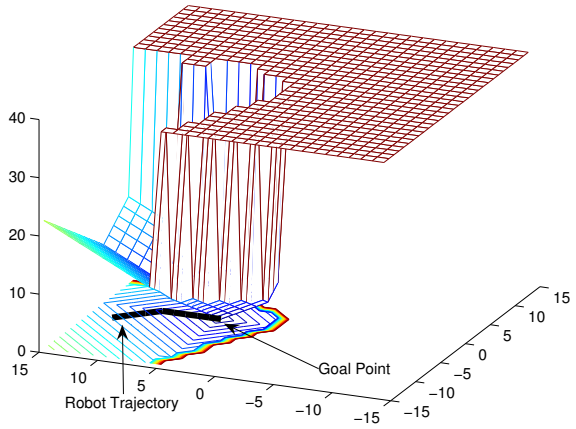

Fig. 4. The plot of the navigation function, it's contours, and the trajectory for the second robot in the upper left hand corner of the region. Notice that only a small "piece" of the obstacle in the upper region intersects the corresponding Voronoi partition.

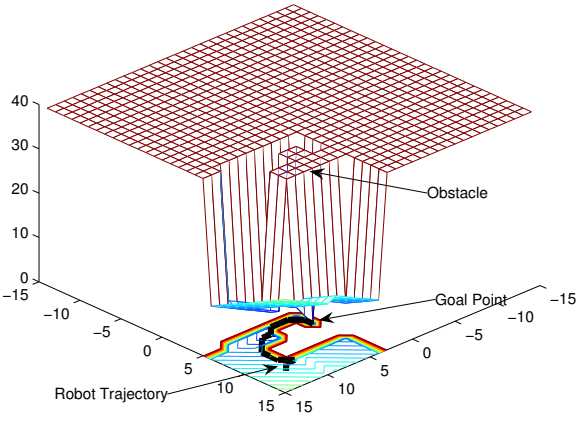

Fig. 5. The plot of the navigation function, it's contours, and the trajectory for the third robot in the upper right hand corner of the region. Notice that not all of the obstacle in the upper region is contained in the third robot's Voronoi partition.

and by NSF grants ECCS CAREER #0811347, IIS #0812338, and CNS #0709329.

REFERENCES

[1] J. C. Latombe, *Robot Motion Planning*. Kluwer, 1991.

[2] J. Cortes, S. Martinez, and F. Bullo, "Spatially-distributed coverage optimization and control with limited-range interactions," *ESIAM: Control, Optimization and Calculus of Variations*, vol. 11, no. 4, pp. 691–719, 2005.

[3] B. Lavis, Y. Yokokohji, and T. Furukawa, "Estimation and control for cooperative autonomous searching in crowded urban emergencies," *IEEE International Conference on Robotics and Automation*, pp. 2831–2836, May 2008.

[4] S. Ferrari, R. Fierro, B. Perteet, C. Cai, and K. Baumgartner, "A geometric optimization approach to detecting and intercepting dynamic targets using a mobile sensor network," *SIAM Journal on Control Optimization*, vol. 48, no. 1, pp. 292–320, 2009.

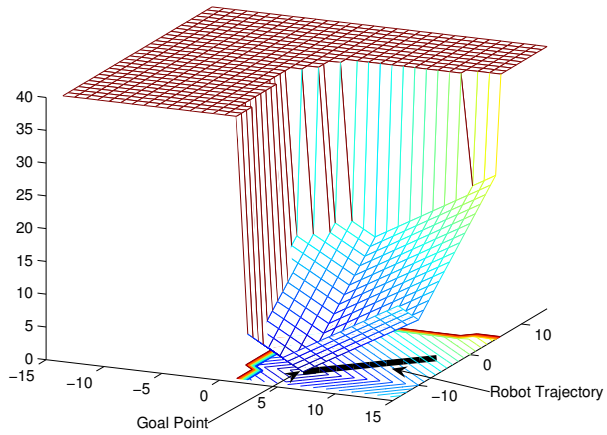

Fig. 6. The plot of the navigation function, it's contours, and the trajectory for the fourth robot which is in the lower right hand corner of the region. The corresponding Voronoi partition does not intersect any obstacles in the region.

[5] R. Cortez, X. Papageorgiou, H. G. Tanner, A. V. Klimenko, K. N. Borozdin, R. Lumia, and W. C. Priedhorsky, "Smart radiation sensor management: Nuclear search and mapping using mobile robots," *IEEE Robotics and Automation Magazine*, vol. 15, no. 3, pp. 85–93, September 2008.

[6] R. Y. Rubinstein, *Simulation and The Monte Carlo Method*. John Wiley and Sons, 1981.

[7] P. Ogren and N. Leonard, "A convergent dynamic window approach to obstacle avoidance," *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 188–195, April 2005.

[8] H. G. Tanner, "Switched UAV-UGV cooperation scheme for target detection," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 3457–3462.

[9] A. Torn and A. Zilinskas, *Global Optimization*. Springer-Verlag, 1987.

[10] D. Koditscheck and E. Rimon, "Robot navigation functions on manifolds with boundary," *Advances in Applied Mathematics*, vol. 11, pp. 412–442, 1990.

[11] H. G. Tanner and A. Kumar, "Towards decentralization of multi-robot navigation functions," in *IEEE International Conference on Robotics and Automation*, 2005, pp. 4132–4137.

[12] S. G. Loizou and A. Jadbabaie, "Density functions for navigation-function-based systems," *IEEE Transactions on Automatic Control*, vol. 53, no. 2, pp. 612–617, March 2008.

[13] A. Ghaffarkhah and Y. Mostofi, "Communication-aware target tracking using navigation functions - centralized case," *International Conference on Robot Communication and Coordination (ROBOCOMM)*, pp. 1–8, 2009.

[14] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerishce Mahematik 1*, pp. 269–271, 1959.

[15] P. Yang, R. A. Freeman, and K. M. Lynch, "Distributed cooperative sensing using consensus filters," *IEEE International Conference on Robotics and Automation*, pp. 405–410, April 2007.

[16] M. Kvasnica, P. Grieder, and M. Baotić, "Multi-Parametric Toolbox (MPT)," 2004. [Online]. Available: http://control.ee.ethz.ch/ mpt/

[17] D. Cruz, J. McClintock, B. Perteet, O. Orqueda, Y. Cao, and R. Fierro, "Decentralized cooperative control: A multivehicle platform for research in networked embedded systems," *IEEE Control Systems Magazine*, vol. 27, no. 3, pp. 58–78, June 2007.

[18] L. C. Pimenta, V. Kumar, R. C. Mesquita, and G. A. S. Pereira, "Sensing and coverage for a network of heterogeneous robots," *IEEE Conference on Decision and Control*, pp. 3947–3852, December 2008.