Actuator Networks for Navigating an Unmonitored Mobile Robot

Jeremy Schiff[§], Anand Kulkarni[†], Danny Bazo[§], Vincent Duindam[§],
Ron Alterovitz[§], Dezhen Song[‡], Ken Goldberg[§†]
[§] Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720-1770, USA
[†] Dept. of Industrial Engineering and Operations Research, University of California, Berkeley, CA 94720-1777, USA
[‡] Dept. of Computer Science, Texas A&M University, College Station, TX, 77843-3112, USA
{jschiff|anandk|dbazo|vincentd|ronalt|goldberg}@berkeley.edu, dzsong@cs.tamu.edu

***Abstract*— Building on recent work in sensor-actuator networks and distributed manipulation, we consider the use of pure actuator networks for localization-free robotic navigation. We show how an actuator network can be used to guide an unobserved robot to a desired location in space and introduce an algorithm to calculate optimal actuation patterns for such a network. Sets of actuators are sequentially activated to induce a series of static potential fields that robustly drive the robot from a start to an end location under movement uncertainty. Our algorithm constructs a roadmap with probability-weighted edges based on motion uncertainty models and identifies an actuation pattern that maximizes the probability of successfully guiding the robot to its goal.**

Simulations of the algorithm show that an actuator network can robustly guide robots with various uncertainty models through a two-dimensional space. We experiment with additive Gaussian Cartesian motion uncertainty models and additive Gaussian polar models. Motion randomly chosen destinations within the convex hull of a 10-actuator network succeeds with with up to 93.4% probability. For n actuators, and m samples per transition edge in our roadmap, our runtime is $O(mn^6)$.

Keywords: Sensor Networks, Actuator Networks, Robotic Navigation, Potential Fields, Motion Planning

I. Introduction

As robots become smaller and simpler and are deployed in increasingly inaccessible environments, we need techniques for accurately guiding robots in the absence of localization by external observation. We explore the problem of observation- and localization-free guidance in the context of *actuator networks*, distributed networks of active beacons that impose guiding forces on an unobserved robot.

In contrast to sensor networks, which passively observe their environment, actuator networks actively induce a physical effect that influences the movement of mobile elements within their environment. Examples include light beacons guiding a robot with an omni-directional camera through the dark, electric fields moving a charged nano-robot through a fluid within a biological system, and radio transmitters on sensor motes guiding a robot with a single receiver.

*This research is supported in part by NSF CISE Award: Collaborative Observatories for Natural Environments (Goldberg 0535218, Song 0534848), Netherlands Organization for Scientific Research (NWO), NIH (F32CA124138), by NSF Science and Technology Center: TRUST, Team for Research in Ubiquitous Secure Technologies, with additional support from Cisco, HP, IBM, Intel, Microsoft, Symmantec, Telecom Italia and United Technologies, and by the AFOSR Human Centric Design Environments for Command and Control Systems: The C2 Wind Tunnel, under the Partnership for Research Excellence and Transitions (PRET) in Human Systems Interaction.

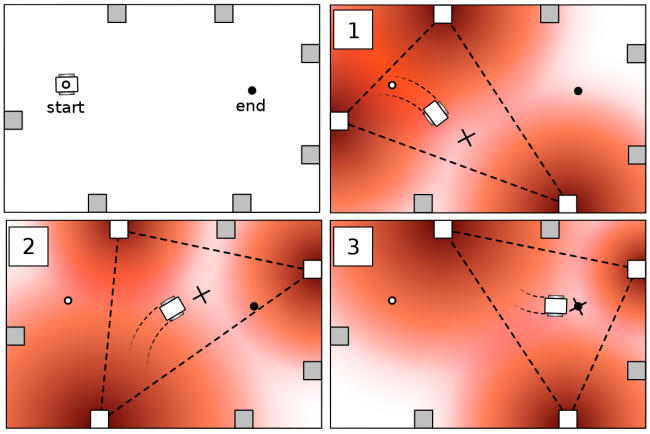


Fig. 1. An actuator network with sequentially activated actuators triplets (shown as squares) driving a mobile robot toward an end location. The robot is guided by creating locally convex potential fields with minima at waypoints (marked by ×s).

We consider a network of beacons which exert a repellent force on a moving element. This repellent effect models many systems for which purely attractive models for beacon-assisted navigation are not realistic or practical. The repulsive effect of an actuator network permits the guided element to pass through specific points in the interior of a region while avoiding contact with the actuators themselves, unlike traditional attractive models for beacon-assisted navigation. Due to their simple structure, actuators can be low-cost, wireless, and in certain applications low-power.

In this paper, we consider a specific application of actuator networks: guiding a simple mobile robot with limited sensing ability, unreliable motion, and no localization capabilities. A motivating scenario for such a system emerges from the potential use of low-cost robots in hazardous-waste monitoring and cleanup applications, such as the interior of a waste processing machine, nuclear reactor, or linear accelerator. Robots in these applications must execute cleanup and monitoring operations in dangerous, contaminated areas where, for safety or practical reasons, it is impossible to place human observers or cameras, or to precisely place traditional navigational waypoints. In this scenario, an actuator network of radio or light beacons may be semi-randomly scattered throughout the region to aid in directing the robot from location to location through the region. By shifting responsibility for navigation to a network of actuators which

directs a passive robot, smaller and more cost-effective robots may be used in these applications.

We consider the 2D case, where the actuator network consists of beacons at different locations in a planar environment. The actuator network is managed by control software that does not know the position of the robot, but it is aware of the positions of all of the actuators. Such software may be either external to the system, or distributed across the actuators themselves, as in the case of a sensor network. We assume only that the robot's sensor is able to detect and respond consistently to the strength and direction of each actuation signal. Such a general framework accurately represents a wide variety of real-world systems in use today involving deployed navigational beacons, while significantly reducing the technical requirements for both the beacons and the robots.

We present an algorithm that sequentially switches between sets of actuators to guide a robot through a planar workspace towards an end location as shown in Figure 1. Using three non-collinear actuators, each one generating a repellent force field with intensity falling off inversely proportional to the square of the distance, we generate potential fields that drive the robot from any position within the convex hull of the actuators to a specific position within the interior of the triangle formed by the actuators (the local minimum of the potential field). The optimal sequence of potential fields to move the robot to a specific point within the interior of the workspace is determined using a roadmap that incorporates the possible potential fields as well as uncertainty in the transition model of the robot. Despite this uncertainty and the absence of position measurement, the locally convex nature of potential fields ensures that the robot stays on track. We present results from simulated actuator networks of small numbers of beacons in which a robot is successfully guided between randomly chosen locations under varying models of motion uncertainty. With as few as 10 actuators in a randomly-placed actuator network, our algorithm is able to steer a robot between two randomly selected positions with probability 93.4% under motion uncertainty of 0.2% standard deviation Gaussian Polar motion uncertainty (perturbing the robot's magnitude and angle with additive Gaussian motion uncertainty).

II. Related Work

There is a long history of investigation into the characteristics of potential fields induced by physical phenomena and how objects are affected by these fields. Some of the earliest work, related to the study of gravitational, electric and magnetic fields, as well as topological properties of such fields, was pioneered by Newton, Gauss, Laplace, Lagrange, Faraday, and Maxwell [1], [2].

Potential functions for robotic navigation have been studied extensively as tools for determining a virtual field which a robotic element can follow to an end location while avoiding obstacles. See Choset et al. [3] for an in-depth discussion. Khatib proposed a model where an end location is represented as an attractor, obstacles are represented as repulsers, and the overall field is determined by the superposition of these fields [4], [5]. While these fields are typically referred to as potential fields, they are actually treated as vector fields that provide the desired velocity vector for the robot at any location in space. While moving, the robot performs simple gradient descent on this space. Motion planning requires defining attractive and repulsive fields such that the robot will always settle at a local minimum created at the end location. Several extensions exist to this classical approach, for example to prevent the robot from getting stuck at a local minimum [6] and to deal with moving obstacles [7], [5].

Rimon and Koditschek addressed this problem by determining the attractive and repulsive potential functions necessary to guarantee unique minima [8]. They accomplished this by defining functions over a "sphere world" where the entire space and all obstacles were restricted to be n-dimensional spheres. They discovered a mapping from this solution to other types of worlds such as "star-shaped worlds", allowing for a general solution to this problem. Connolly et al. [6] use harmonic functions in the repulsive and attractive functions to avoid local minima. In our work, we also address the problem of local minima, but are restricted in our choice of potential functions.

Whereas past research has treated modeling a robot's environment and guidance instructions as an arbitrary virtual potential function, our work considers a network of actuators which emit signals from given, fixed positions and explicitly models physical phenomena. In our formulation, these actuators are the sole contributors to the potential field, rather than the environment or guidance information. In addition, the potential fields in our algorithm transition at discrete time steps, and problems related to local minima are avoided by steering the robot through waypoints, rather than directly from start to end location.

The concept of distributed actuation has been studied in various contexts. Li et al. [9] examined how to directly apply potential functions in a distributed fashion over sensor networks, focusing on formulating the algorithm in a distributed manner. Pimenta et al. [10] addressed the robot navigation problem by defining force vectors at the nodes of a graph. They assume, however, that nodes can be arbitrarily added to the graph, contrary to our problem formulation in which the actuators are provided as input and are fixed. Finally, research in distributed manipulation has examined how to leverage many actuators to perform coordinated manipulation, focusing primarily on the use of vibratory fields to place and orient parts [11], [12], [13], [14], [15].

To actively construct potential fields to steer a mobile robot subject to uncertainty in motion and sensing, we build on previous results in motion planning under uncertainty [16]. Motion planners using grid-based numerical methods and geometric analysis have been applied to robots with motion and sensing uncertainty using cost-based objectives and worst-case analysis [17], [18], [16]. Markov Decision Processes have been applied to motion planning with uncertainty in motion, but these methods generally require state sensing and are not directly applicable to actuator networks [19],

[20], [21], [16].

In this paper, we consider a hybrid sensing uncertainty model: the actuator network cannot sense the mobile robot but the mobile robot can sense the potential field. Lazanas and Latombe proposed a landmark-based approach in which the robot has improved sensing and actuation inside landmark regions, reducing the complexity of the motion planning problem to moving between these regions [22]. An actuator in an actuator network can be viewed as a generalization of a landmark that can be controlled and can exert influence over the mobile robot at arbitrary distances. As with sensing uncertainty, this paper considers a hybrid actuation uncertainty model: the actuator network generates a precise potential field with no uncertainty while the mobile robot is subject to uncertainty in its motion. To address motion uncertainty, Alterovitz et al. introduced the Stochastic Motion Roadmap (SMR), a sampling-based method that explicitly considers models of motion uncertainty to compute actions that maximize the probability that a robot will reach an end location [21]. As in SMR, we use the objective of maximizing probability of success over a roadmap. But unlike SMR, which assumes perfect sensing, the maximum probability path for an actuator network must be computed before plan execution is begun since sensing feedback is not available.

III. Problem Formulation

A. Assumptions

We consider the control of a single mobile robot in a planar environment, influenced by an actuator network. The n actuators are located at known positions $\mathbf{x_i} \in \mathbb{R}^2$. Each actuator can be controlled independently to produce a signal with piece-wise constant amplitude $a_i(t) \geq 0$. Each actuator generates a radially symmetric potential field U_i of the form

$$U_i(\mathbf{x}) = \frac{a_i}{|\mathbf{x} - \mathbf{x_i}|} \tag{1}$$

The direction and magnitude of the gradient of this field can be observed from any location $\mathbf{x} \in \mathbb{R}^2$ and are given by the vector field $\mathbf{F_i}$

$$\mathbf{F_i}(\mathbf{x}) = \frac{\partial U_i}{\partial \mathbf{x}} = -\frac{a_i(\mathbf{x} - \mathbf{x_i})}{|\mathbf{x} - \mathbf{x_i}|^3} \tag{2}$$

i.e. the signal strength $|\mathbf{F_i}(\mathbf{x})|$ is inversely proportional to the square of the distance to the actuator, as is common for physical signals.

The aim of the actuator network is to guide a mobile robot along the direction of steepest descent of the combined potential field $U(\mathbf{x}) = \sum_i U_i(\mathbf{x})$. The position of the robot as a function of time is denoted by $\mathbf{p}(t) \in \mathbb{R}^2$, and the robot is assumed to have sufficient local control to be able to move approximately in a given direction vector $\mathbf{v}$ (relative to its own coordinate frame). The desired motion direction $\mathbf{v}$ is equal to the direction of steepest descent of the combined potential field U and is hence given by the vector sum

$$\mathbf{v} = -\sum_i \mathbf{F_i}(\mathbf{x}) = \sum_i \frac{a_i(\mathbf{x} - \mathbf{x_i})}{|\mathbf{x} - \mathbf{x_i}|^3} \tag{3}$$

Under this control strategy, each of the actuators serves effectively as a "robot repulser", causing the robot's direction of motion to be determined by the total combined force from all actuators. We only assume that the robot can continuously measure the direction and strength of the superimposed actuator signals. In particular, no global position sensors or odometry is required. Its motion is described by a general model $\dot{\mathbf{p}} = R(\mathbf{v})$, which may include stochastic components to represent sensor and actuator uncertainty, or general uncertainty in movement across the workspace due to viscosity in a fluid environment, uneven terrain causing wheel slip, or simply unreliability in the robot's design.

The design of a suitable sensor for the robot to measure the actuator signals depends on the type of actuation used. For light-based actuation, one could use an omni-directional camera. Each actuator signal from a certain direction causes a specific spot in the image to light up with a brightness depending on the actuator strength and distance. The desired motion vector $\mathbf{v}$ can then be computed simply by adding the force directions corresponding to all image pixels, weighted by their brightness.

A centralized controller is assumed to know the location of the actuators and to be able to control the actuator amplitudes in a time-discrete manner. It does not have other sensing information; in particular, it cannot measure the robot's position.

B. Inputs and Output

The inputs of the control algorithm are the initial location $\mathbf{p_0} = \mathbf{p}(0) \in \mathbb{R}^2$, the end location $\mathbf{p_e} \in \mathbb{R}^2$ of the robot, and the locations of each actuator $\mathbf{x_i} \in \mathbb{R}^2$. The actuator locations can either be determined a priori, or via some localization scheme. The key aspect is that the actuators will not observe or track the robot.

The proposed algorithm returns a sequence of actuator amplitudes $\{a_i(t_j)\}$ at discrete time instants t_j that maximizes the probability that the robot successfully moves from the start location $\mathbf{p_0}$ to the end location $\mathbf{p_e}$. If no path from $\mathbf{p_0}$ to $\mathbf{p_e}$ can be found, the algorithm returns the empty sequence. Each set of amplitudes contains exactly three nonzero values corresponding to a particular triangle of actuators and an associated start and destination location. The use of only three actuators at a time has the advantage of simplifying the analysis of the system and potentially reducing power consumption of the network by limiting the number of active actuators at any time. To conserve power, all idle actuators can go into a low-power state. Each time instance is separated by a sufficiently large duration that a robot starting at the associated start location will by this time either have migrated to the associated target location moving at minimum velocity or will be outside the actuator triangle and get progressively farther away. This time is determined by the minimum speed of the robot.

C. Motion Uncertainty

To investigate the utility of actuator networks in steering robots with *uncertain* motion, we considered two uncertainty

models $R(v)$: one using additive random Gaussian noise on the robot's Cartesian coordinates, and one using additive random Gaussian noise on the robot's polar coordinates. In other words, the Cartesian motion uncertainty model describes the uncertain motion of the robot as

$$\begin{bmatrix}\dot{p}_x \\ \dot{p}_y\end{bmatrix} = \dot{\mathbf{p}} = R(\mathbf{v}) = \begin{bmatrix}N\left(v_x, \sigma^2\right) \\ N\left(v_y, \sigma^2\right)\end{bmatrix} \tag{4}$$

while the polar motion uncertainty model is given by

$$\begin{bmatrix}\dot{p}_x \\ \dot{p}_y\end{bmatrix} = \dot{\mathbf{p}} = R(\mathbf{v}) = \begin{bmatrix}r\cos(\theta) \\ r\sin(\theta)\end{bmatrix} \tag{5}$$

where r and θ are drawn from Gaussian distributions as

$$r = N\left(|\mathbf{v}|, \sigma^2\right) \qquad \theta = N\left(\text{atan2}(v_y, v_x), \frac{\sigma^2}{4\pi^2}\right)$$

IV. Motion control using actuator networks

To solve the problem of finding a valid actuation sequence (if one exists), the algorithm first generates all possible $\binom{n}{3}$ triangles that can be formed using the n actuators. Then, it computes the incenters of these triangles (as discussed in Section IV-A) to be used as local minima of the potential fields. These incenters define the vertices in a graph, and the next step of the algorithm is to determine the weights of all possible edges between the vertices. We define the weight of an edge to be the probability that the robot successfully navigates from one vertex to the other, as defined by the robot motion model $R(\mathbf{v})$. The resulting graph defines a roadmap for the robot, and the last step of the algorithm is to insert the start and end locations into the roadmap and determine the path between them that maximizes the probability of successfully reaching the end location.

A. Local control using actuators triplets

The following assumes the actuators we choose are not collinear. Consider a potential field U generated by an actuators triplet i, j, and k (and all the other actuators in the network set to zero amplitude). If the amplitudes of the actuator triplet is strictly positive, the potential field will have a local minimum at some point inside the triangle. This location is called the *waypoint* of the potential field. In our global algorithm, we may traverse many waypoints to get from the start to the end location. The final waypoint is defined to be the end location. For a given triangle structure, we define the *feasible region* as the set of locations that can be made waypoints, that is, local minima of the potential field. The feasible region is clearly strictly smaller than the triangle defined by the active actuators.

For a given waypoint $\bar{\mathbf{x}}$, $C(\bar{\mathbf{x}})$ is the *capture region* of a waypoint as the set of all points $\mathbf{x} \in \mathbb{R}^2$ such that, when following the direction of steepest descent from $\mathbf{x}$, one will eventually arrive at $\bar{\mathbf{x}}$. Some examples of capture regions for various triangles and waypoints are given in Figure 2.

For a point $\bar{\mathbf{x}}$ to be a local minimum of the potential field, the gradient $\frac{\partial U}{\partial \mathbf{x}}$ at $\bar{\mathbf{x}}$ should be zero, and the Hessian matrix

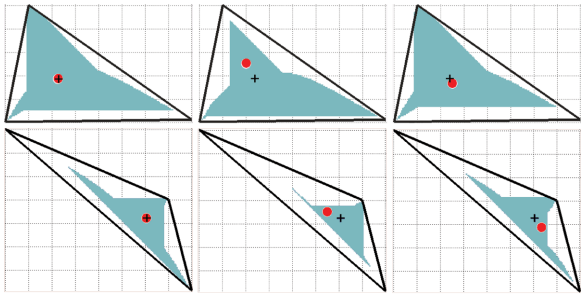


Fig. 2. Examples of two different triangles and three different waypoints (denoted as circles) and their capture regions (shaded areas) for both triangles. The incenter of each triangle is marked with a '+'.

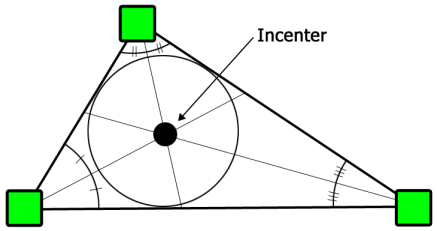


Fig. 3. Example of a triangle with its incenter.

$\frac{\partial^2 U}{\partial \mathbf{x}^2}$ should be positive definite. This means that the actuator amplitudes $a_i, a_j, a_k > 0$ must be chosen such that

$$\frac{\partial U}{\partial \mathbf{x}}(\bar{\mathbf{x}}) = \begin{bmatrix}\frac{\mathbf{x_i}-\bar{\mathbf{x}}}{|\bar{\mathbf{x}}-\mathbf{x_i}|^3} & \frac{\mathbf{x_j}-\bar{\mathbf{x}}}{|\bar{\mathbf{x}}-\mathbf{x_j}|^3} & \frac{\mathbf{x_k}-\bar{\mathbf{x}}}{|\bar{\mathbf{x}}-\mathbf{x_k}|^3}\end{bmatrix} \begin{bmatrix}a_i \\ a_j \\ a_k\end{bmatrix} = \mathbf{0} \tag{6}$$

The space of a_i satisfying Equation 6 is a one-dimensional vector space, but since both Equation 6 and the signature of the Hessian are scale invariant, checking whether $\bar{\mathbf{x}}$ is in the feasible region reduces to 1) checking whether the elements of any solution vector a_i of Equation 6 have equal sign (all positive or all negative), and if so 2) checking whether for a choice of positive a_i the Hessian at $\bar{\mathbf{x}}$ is positive definite.

In the global control law described in the following section, we choose a specific point in each triangle to be the waypoint, namely the *incenter*. The *incenter* of a triangle is the center of its inscribed circle, or equivalently, the intersection of the three angle bisectors of the triangle's vertices (Figure 3). In extensive simulation of many different triangles shapes, the incenter was always found to be in the feasible region, and when chosen as the waypoint, the capture region of the incenter was found to generally be larger than other centers, including the centroid. While the incenter has performed well, other ways of determining waypoints can be considered. A formal proof of these favorable properties of the incenter is the subject of future work.

B. Global control using switching potential field

To extend the previously described static local control law from an actuator triplet to a full actuator network, we define a roadmap that robustly guides the robot from its start location, via the incenters of successive triangles defined by actuator

Algorithm 1 The Actuator Network Algorithm

```
triangles ← computeTriangles(actuatorLocations)
vertices ← computeIncenterLocations(triangles)
graph ← computeEdgeWeights(vertices)
path ← computePath(startVertex, endVertex, graph)
```

triplets in the network, to its end location. The steps in the algorithm are as follows (Algorithm 1):

1) Compute all $\binom{n}{3}$ triangles that can be generated by the n actuators in the network.
2) Compute the incenters of the triangles and designate these incenters as vertices in a graph.
3) For every pair of vertices (v_1, v_2) in the graph, add a directed edge from v_1 to v_2 if v_1 is in the capture region of the potential field with local minimum at v_2. Use the robot motion model $R(\mathbf{v})$ to compute the probability $P(v_2|v_1)$ that the robot moves from v_1 to v_2 in this potential field. Set the edge weight to be the negative log of this probability: $-\log P(v_2|v_1)$.
4) The weighted graph forms a roadmap for the robot. Add the start location and end location to the graph and run Dijkstra's algorithm [16] to obtain the optimal path from start to goal, or, if no such path exists, the empty sequence.

The resulting shortest path is a sequence $\{v_i\}$ of vertices, or, with the exception of the start and end location, a sequence of incenters. We can robustly drive the robot from incenter to incenter by successively switching the amplitudes such that the next incenter v_{i+1} in the path becomes the new waypoint. Since the point v_i is in the capture region of v_{i+1}, the robot will be driven to the end location. Even though no position sensing mechanism for the robot is used, and even though the robot model contains stochastic components, the convergent nature of the potential fields will ensure that the position motion uncertainty does not grow unbounded. As long as the actuators do not move, step 4 can be repeated using the same roadmap to solve multiple queries for different start and end locations.

C. Computational Complexity

In step 1 with n actuators, we explore $O(n^3)$ triangles. For step 2, it takes O(1) time to compute an incenter location and associated actuator amplitudes, thus it takes $O(n^3)$ to complete this step. For step 3 with m rejection samples per edge, there is an edge for each pair of incenters, thus this step will take $O(mn^6)$. For step 4, there are $O(n^3)$ vertices and $O(n^6)$ edges. Because Dijkstra's algorithm takes $O(|E| + |V|\log|V|)$, step 4 takes $O(n^6)$. Thus, the total runtime is $O(mn^6)$.

While this result is polynomial in m and n, for certain applications requiring very fast construction of actuation strategies for very large actuator networks, it may be desirable to further reduce this runtime. The most immediate way to reduce the runtime is to not consider all possible $\binom{n}{3}$ triangles. Instead, we can modify step 1 in Section IV-B only use triangles of reasonable size (not too small or too large) or discard triangles for which the capture region is fully contained in the capture region of other triangles.

It is also important to note that the computation of the roadmap is an *offline* procedure that must be carried out only once during the preparation of the roadmap for a given actuator network. In addition, many implementations of an actuator network may self-correct for an increased number of actuators by providing additional resources for computation along with each actuator. The (notably parallel) problem of computing edge weights could be solved using distributed computation across the computation elements.

D. Implementation aspects

To compute the motion probabilities for the edge weights in step 3 of the algorithm, we perform rejection sampling with m samples. For an edge from v_1 to v_2, we compute the robot motion from v_1 by integrating the robot velocity $R(\mathbf{v})$ using Euler integration. If, after a certain integration interval τ the robot is within some small distance ϵ of v_2, we consider the motion successful, and failure otherwise. Thus, each edge's weight is determined by the percentage of successful transitions from v_1 to v_2. Valid values for ϵ depend on the specific robot's size relative to the workspace and sensitivity to motion uncertainty, while τ depends strictly on the size of the region and the natural velocity of the robot. For instance, τ may be defined as the maximum amount of time required for a robot to move linearly between any two points in the planar region at its minimum velocity with no motion uncertainty, and ϵ may be set to 1% of the minimum distance between any two actuators. Tighter bounds are possible for faster movement. Even though v_1 may be in the capture region of v_2, sensor and actuator uncertainty (as captured by $R(\mathbf{v})$) can cause the robot to move temporarily outside the capture region, after which it will diverge and not succeed in reaching the current waypoint. The probability of success is determined by the fraction of the samples that successfully reach the waypoint.

The weights of the edges are taken to be the negative logarithm of the probability of success. The probability of successfully reaching the end location along a certain path is equal to the product of the probabilities of successfully moving along the edges of the path. Since multiplying probabilities P_i is equivalent to adding log-probabilities $\log P_i$, maximizing the probability of success along a path is equivalent to minimizing the sum of the negative logs of the probabilities along a path. Thus, we can efficiently compute the path with the maximum probability by using Dijkstra's algorithm from $\mathbf{p_0}$ to $\mathbf{p_e}$ using the negative log of the probability at each edge.

V. Simulation experiments

All simulation experiments were implemented in Matlab and executed on PCs with a 2.0GHz Intel processor and 2GB of RAM.

We fixed the workspace to be 5×5 units, and used $m = 10$ rejection samples. We explored the algorithm's effectiveness

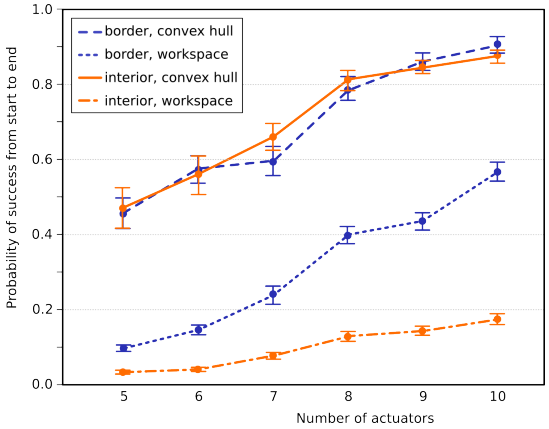


Fig. 4. Simulation result averaging over 100 trials of an actuator network and 100 start-end location pairs, with $n = 5 \ldots 10$ actuators and two placement strategies: border placement and interior placement. Results for each are shown with start and end locations chosen randomly across the entire workspace or only within the convex hull of the actuators. These simulations used 0.01 standard deviation polar motion uncertainty.

under a variety of actuator location distributions (Section V-A) and motion uncertainty models (Section V-B). For every choice of actuator network and robot uncertainty model, we randomly chose $k = 100$ start and end locations in the workspace, computed the optimal paths using Algorithm 1, and simulated the motion of the robot in the actuator network.

For a given actuator network topology, we compute the average probability of the robot successfully traveling from a random start location to a random end location.

A. Varying Actuator Placement

We randomly placed $n \in \{5, \ldots, 10\}$ actuators throughout the workspace according to two distribution models. In the *bordered* distribution strategy, each actuator was placed at a location chosen uniformly at random on the border of the workspace. In the *interior* distribution strategy, actuators were scattered uniformly at random throughout the entire workspace. For each actuator placement model and for each possible number of actuators from 5 to 10, 100 random actuator geometries were produced. The robot motion model was set to have zero motion uncertainty and a probabilistic roadmap was constructed according to the above algorithm.

The results are shown in Figure 4. We examine the two cases where 1) the start and end locations are within the convex hull of the actuators and 2) the start and end locations are anywhere in the workspace. In the first case, the data suggests that the border selection strategy performs just as well as the interior selection strategy. Because the convex-hull eliminates start/end locations that are outside the convex hull and therefore impossible to reach, we can see that there is no robustness advantage for one method over another. As the algorithm performs comparably in the convex-hull restricted test, the border-selection method is better in the full-workspace experiment, because on average, its convex hull will cover a larger area, which in turn means more start/end locations will be reachable.

As would be expected, increasing the number of actuators increases the probability of success with the workspace model. We can see that the probability of success also improves as we increase actuators for the convex-hull method. We discuss this property further in Section V-B.

The simulation results suggest lower bounds on the effectiveness of smart actuator placement strategies. Better results can be obtained by (deterministically) optimizing the placement of actuators to 1) maximize the area of the workspace that falls into the capture region of at least one triangle, and 2) maximize the connectivity between points in the workspace, particularly when likely start and end locations are known in advance. Such an optimal-placement algorithm will be the subject of future research.

B. Varying Motion Uncertainty

For networks of $n \in \{5, \ldots, 10\}$ actuators, we examined how the probability of successful completion varied for both the Cartesian motion uncertainty model described in Equation 4 and the polar motion uncertainty model described in Equation 5. We experiment with different errors (standard deviations $\sigma \in \{0, 0.01, 0.05, 0.1, 0.2\}$). The results are summarized in Figure 5, and Figure 6 shows an example of the roadmap generated by the algorithm.

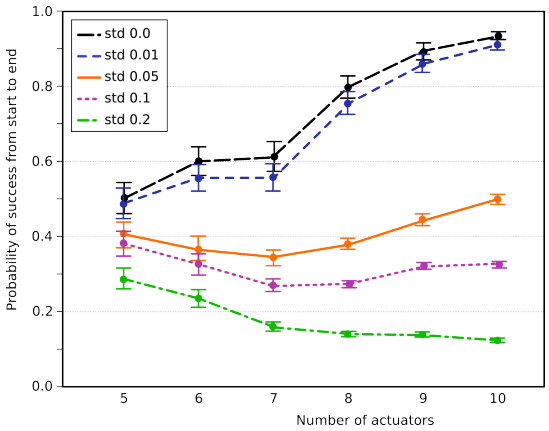
The figure shows that increasing the motion uncertainty will result in reduced probability of success, for both motion uncertainty models and any number of actuators. Under large motion uncertainty, the robot is more likely to drift outside the capture region, resulting in failure to reach the next waypoint and hence to successfully complete the path to the end location.

Because these examples are only of start/end locations within the convex hull, adding actuators has two effects. As we increase the number of actuators, the area of the convex hull will become larger, which means that the average path length between start/end goals in the convex hull increases, making the effects of the motion uncertainty more significant. More actuators also means more flexibility in the number of paths, due to an increased number of incenters and overlapping triangles. As we increase the number of actuators, the incremental addition to the convex hull will decrease, and the number of additional waypoints will grow quadratically with the number of new actuators. Thus, for larger motion uncertainty models, the probability of success first decreases and then increases. This effect is more extreme depending on how significant the motion uncertainty is.

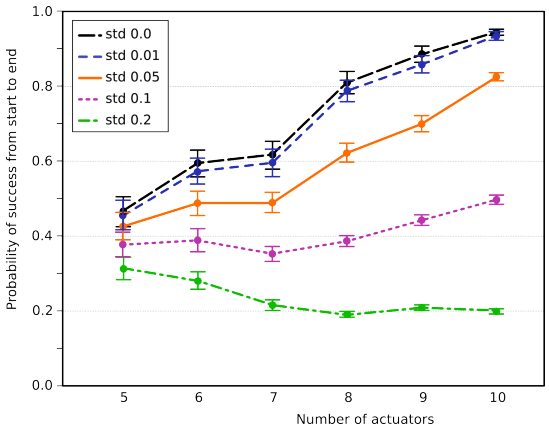
For 10 actuators, standard deviations of 0.0, 0.01, 0.05, 0.1, and 0.2, and Cartesian motion uncertainty, the average probabilities of success are 94.5%, 93.4%, 82.5%, 49.6%, and 19.8%. For Polar motion uncertainty, the average probabilities of success are 93.3%, 91.0%, 49.8%, 32.4%, and 12.2%.

VI. Conclusions and Future Work

We consider the problem of localization-free guidance of a robot using an actuator network of beacons for use in steering simple, low-cost robots. The Actuator Networks system and

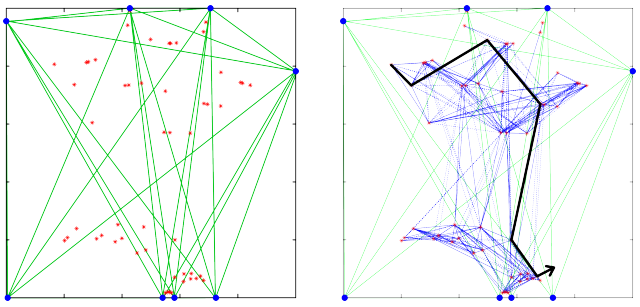


(a) Average success rate under Cartesian motion uncertainty.



(b) Average success rate under polar motion uncertainty.

Fig. 5. Comparison of the results of varying Gaussian motion uncertainty for a fixed set of actuator locations under two motion uncertainty models, with start and end locations chosen within the convex hull of the actuators.



(a) Triangles and corresponding incenters (denoted $*$) generated from 8 randomly placed actuators.

(b) Roadmap showing edges between incenters and an example path (thick line) through the network.

Fig. 6. Example of a simulation of the actuator-networks algorithm with $n = 8$ actuators and Cartesian motion uncertainty with $\sigma = 0.01$. The actuators are placed randomly on the border of a square workspace, the incenters of all possible triangles between them form vertices in a roadmap with edges containing the probability of successful transition by activation of an actuator triplet.

algorithm steer unmonitored robots between points using an external network of actuators and a probabilistic roadmap. Our algorithm was able to produce relatively high probabilities of successful navigation between randomly-selected points even in the presence of motion uncertainty.

The low number of actuators necessary for successful steering in our technique has important consequences for the robustness of these methods in practice. An inexpensive way to guarantee continuous operation of an actuator network is to use more than the minimum number of actuators required for high-probability performance; as an example, with 20 actuators under border placement and 1% Cartesian motion uncertainty, even if half of the actuators eventually fail, the probability of completion would not drop significantly.

We plan to explore several extensions in future work. The technique of actuator networks can be extended to consider obstacles in the workspace. An obstacle can affect an Actuator Network in three ways: (1) the obstacle restricts the motion of the mobile robot in the workspace, (2) the obstacle blocks the signal from an actuator, and (3) the obstacle causes multi-path effects as the signals from the actuators reflect off the obstacles. To model case 1, we can represent obstacles implicitly in the graph via the edge weights encoding transition success probabilities. As discussed in Sec. IV-D, we estimate the probability $P(v_2|v_1)$ of successfully moving from vertex v_1 to vertex v_2 by simulating the robot's motion as it follows the gradient of the signal generated by the actuators. If the mobile robot's motion intersects an obstacle during a simulation, we determine that a failure has occurred in our rejection sampling step. For case 2, we can modify the simulation so the gradient used by the mobile robot does not include signal from a particular actuator if a line segment between that actuator and the mobile robot's current location intersects an obstacle. Since the probability of success for edges in the graph will decrease when obstacles are present, the number of actuators necessary in order to find a feasible plan will increase. Case 3 is known to be difficult to model effectively, and is a significant problem for certain domains such as RSSI localization. As future work, we can also evaluate how different multi-path models will affect the robot by including this in the determination of the edge-weights in a similar fashion as case 2.

We would also like to explore the alternative problem of designing an algorithm for placement of actuators.

VII. Acknowledgments

We thank the members of the UC Berkeley Automation Sciences Lab for their feedback and support, including particularly helpful contributions from Ephrat Bitton, Jijie Xu, and Menasheh Fogel. We also thank Claire Tomlin, and Shankar Sastry for their support.

References

[1] O. D. Kellogg, "Foundations of potential theory," 1969.
[2] J. C. Maxwell, "On hills and dales," *The Philosophical Magazine*, vol. 40, no. 269, pp. 421–427, 1870.
[3] H. Choset, K. M. Lynch, S. Hutchingson, G. Kantor, W. Burgand, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*, 1st ed. MIT Press, 2005.
[4] O. Khatib, "Commande dynamique dans l'espace opérationnel des robots manipulateurs en présence d'obstacles," Ph.D. dissertation, École Nationale Supérieure de l'Aéronautique et de l'Espace, Toulouse, France, 1980.

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

[6] C. Connolly, J. Burns, and R. Weiss, "Path planning using Laplace's equation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, May 1990, pp. 2102–2106.

[7] W. S. Newman and N. Hogan, "High speed robot control and obstacle avoidance using dynamic potential functions," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1987, pp. 14–24.

[8] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robotics and Automation*, vol. 8, no. 5, pp. 501–518, 1992.

[9] Q. Li, M. D. Rosa, and D. Rus, "Distributed algorithms for guiding navigation across a sensor network," in *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM Press, 2003, pp. 313–325.

[10] L. C. A. Pimenta, G. A. S. Pereira, and R. C. Mesquita, "Fully continuous vector fields for mobile robot navigation on sequences of discrete triangular regions," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007, pp. 1992–1997.

[11] K. F. Böhringer and H. Choset, *Distributed Manipulation*, 1st ed. Kluwer Academic Publishers, 2000.

[12] K.-F. Bhringer, V. Bhatt, B. R. Donald, and K. Goldberg, "Algorithms for sensorless manipulation using a vibrating surface," *Algorithmica*, vol. 26, no. 3, pp. 389–429, April 2000.

[13] A. Sudsang and L. Kavraki, "A geometric approach to designing a programmable force field with a unique stable equilibrium for parts in the plane," in *Proceedings of the IEEE Interational Conference on Robotics and Automation*, vol. 2, 2001, pp. 1079–1085.

[14] F. Lamiraux and L. E. Kavraki, "Positioning of symmetric and non-symmetric parts using radial and constant fields: Computation of all equilibrium configurations," *International Journal of Robotics Research*, vol. 20, no. 8, pp. 635–659, 2001.

[15] T. H. Vose, P. Umbanhowar, and K. M. Lynch, "Vibration-induced frictional force fields on a rigid plate," in *Proceedings of the IEEE International Conference on Robotics and Automation*, April 2007, pp. 660–667.

[16] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006.

[17] B. Bouilly, T. Siméon, and R. Alami, "A numerical technique for planning motion strategies of a mobile robot in presence of uncertainty," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, Nagoya, Japan, May 1995, pp. 1327–1332.

[18] S. M. LaValle and S. A. Hutchinson, "An objective-based framework for motion planning under sensing and control uncertainties," *International Journal of Robotics Research*, vol. 17, no. 1, pp. 19–42, Jan. 1998.

[19] T. Dean, L. P. Kaelbling, J. Kirman, and A. Nicholson, "Planning under time constraints in stochastic domains," *Artificial Intelligence*, vol. 76, no. 1-2, pp. 35–74, Jul. 1995.

[20] D. Ferguson and A. Stentz, "Focussed dynamic programming: Extensive comparative results," Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-04-13, Mar. 2004.

[21] R. Alterovitz, T. Siméon, and K. Goldberg, "The Stochastic Motion Roadmap: A sampling framework for planning with Markov motion uncertainty," in *Robotics: Science and Systems*, 2007.

[22] A. Lazanas and J. Latombe, "Motion planning with uncertainty: A landmark approach," *Artificial Intelligence*, vol. 76, no. 1-2, pp. 285–317, 1995.