Motion Planning for Serpentine Robots

Wade Henning†, Frank Hickman†, Howie Choset†

Abstract. *Serpentine robots offer advantages over traditional mobile robots and robot arms because they provide enhanced flexibility and reachability, especially in convoluted environments. These robots are well suited to inspect large space-fairing truss structures such as the future space station and can also be used to inspect the Space Shuttle cargo bay before launch. Serpentine mechanisms offer unique capabilities on Earth to applications such as search and rescue, surface coating, and minimally invasive surgery.*

The work, described in this paper, will exploit a geometric structure, termed a roadmap, to guide the motions of a serpentine robot in highly convoluted spaces. This approach offers advantages over previous work with serpentine robots because it provides a general mathematical structure that is not mechanism specific, thereby having applications to a large class of problems.

1 Introduction

We present some preliminary experimental results in the area of *global sensor based planning* for *hyper-redundant robot manipulators.* Recall from [5] that a hyper-redundant mechanism is a kinematically redundant manipulator in which the degree of redundancy is very large or infinite. Such robots are analogous in morphology to tentacles or snakes. *Sensor based planning* incorporates sensory information into some stage of a robotic motion planning, whether it be navigation, locomotion, grasping, etc. Based on information gathered from sensors, global sensor based planning incrementally defines most of the robot's path and constructs a model of the robot's environment. This differs from *local sensor based planning* [18] which modifies the robot's plan over a span which is short in time or distance, without affecting a world model.

Due to their high degree of articulation, hyper-redundant robots are potentially superior for operations in highly constrained and unusual environments encountered in applications such as inspection of nuclear reactor cores, chemical sampling of buried toxic waste, and medical endoscopy. Hyper-redundant robots can also be used as tentacle-like grasping devices for capturing and manipulating floating satellites [4] or to enable complex "whole arm manipulation." Mobile hyper-redundant robots also offer novel means for locomotion ([7], [10], [9], [1]) in complex environments.

The aforementioned applications are characterized by environments which are difficult to precisely model, and thus global sensor based planning schemes are vital to the realistic deployment of hyper-redundant robots in these applications. In this paper, we describe a global sensor based planning heuristic which combines the hyper-redundant kinematic analysis found in [5], [7], [6], [8] and [2] with the provably correct sensor based exploratory algorithms found in [11], [12] and [13]. These sensor based motion planning schemes are based on a structure, termed the *Generalized Voronoi Graph.*

†Carnegie Mellon University, Pittsburgh, PA

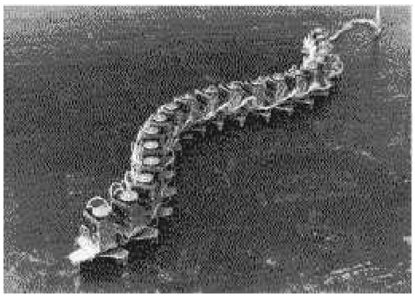

Fig. 1. Hirose Active Chord

2 Prior Work

2.1 Famous Snake Robots

Robotics engineers have already produced serpentine robots for various applications. Professor Hirose at the Tokyo Institute of Technology built the first serpentine robot (Figure 1) and studied how such mechanisms can locomote in the plane. Research serpentine robots in the United States began with the hyper-redundant manipulator built at Caltech by Chirikjian and Burdick (Figure 2) who developed some novel end effector placement algorithms for these robots. They also introduced some new snake robot locomotion theory, as well. Takanashi of NEC developed a serpentine robot for the purposes of search and rescue (Figure 3) for survivors in collapsed buildings. In cooperation with Takanashi, engineers at the Jet Propulsion Laboratory adapted the NEC design and built the JPL Serpentine Robot for use in space station inspection.

2.2 Motion Planning

Sensor based planning has received increased attention, as it is a requirement for realistic deployment of autonomous robots in unstructured environments. Our approach is to adapt the structure of a rigorous motion planning scheme to a sensor based implementation; one such scheme is based on a geometric structure, termed a *roadmap* [15]. Roadmaps are defined by the following

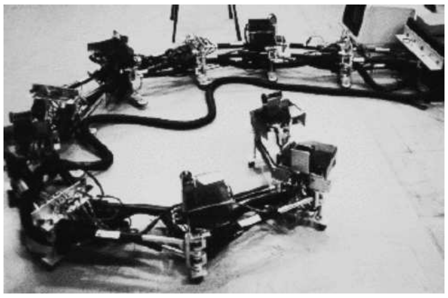

Fig. 2. Caltech Snake Robot

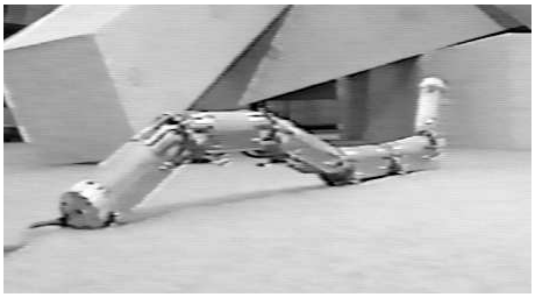

Fig. 3. NEC Search and Rescue

properties: *accessibility*, *departability* and *connectivity*. These properties imply that the planner can construct a path between any two points in a connected component of the robot's free space by first finding a path onto the roadmap (accessibility), traversing the roadmap to the vicinity of the goal (connectivity), and then constructing a path from the roadmap to the goal (departability).

A roadmap called the *Generalized Voronoi Graph* (GVG) has been introduced in [11], [12] and [13]. A key feature of the GVG is that it is one-dimensional in an arbitrary dimensioned space. In the plane, the GVG reduces to the *Generalized Voronoi Diagram* [16] because both structures are one-dimensional. However, the GVG is amenable to sensor based construction because there already exists a well defined incremental construction technique for the GVG; the incremental construction technique is an adaptation of a numerical curve tracing algorithm [13].

There has been little work devoted explicitly to serpentine motion planning. One approach is based on the defenition of *tunnels* through an obstacle field, into which the manipulator "slithers" [3], [2]. Recently, a local sensor based planning method was reported in [18] in which the snake robot maintains its end effector location while it locally adapts to a time varying environment. In this method, the entire manipulator is fit within a tunnel and then part of the tunnel is continuously adapted away from any object that becomes unacceptably close.

One of the first global sensor based planning techniques for highly redundant robots is based on a *tactrix* [17]. This approach, however, assumes that there are perfect sensors on the robot and has not been implemented on a real robot. Finally, none of the aforementioned procedures can be used for mapping an environment with a robot snake.

In this work, we used the results of the GVG approach to construct, in a sensor based fashion, tunnels which have a dual purpose: (1) to prescribe a collision-free path for a hyper-redundant manipulator and (2) to serve as a map of an environment that captures all important topological features..

3 Background

Controlling serpentines presents a difficult challenge. At this point in time there are no adequate control and motion planning strategies which can be used to perform complicated spatial tasks such bridge truss and space station inspection. The proposed work addresses this challenge, which is difficult because serpentine robots, unlike conventional robots, have many degrees of freedom to control. Conventional robots can reach a particular location with one or two configurations, i.e., one or two sets of joint parameters can specify the location and orientation of the end effector of a conventional robot.

However, in a highly convoluted environment such as truss or rubble pile, a conventional robot will not be able to reach all positions without hitting an object. A serpentine robot can uniquely obtain these end effector locations by flexing around multiple beams and columns. In order to do so the serpentine must progressively negotiate into and around the three-dimensional obstacles to get to its final location. In other words, the serpentine robot is no longer just reaching a desired location, but it is following a *path* to get there. This path is a successive set of configurations which bring the robot from an initial configuration to a final configuration with a desired end effector position and orientation. An important property of this motion is that the the body of the serpentine robot follows the path traced by the head.

This work draws from two sub-fields in motion planning: sensor based planning for robots and motion planning for hyper-redundant manipulators, both of which are briefly reviewed in this section.

3.1 Hyper-redundant Manipulator: Inverse Kinematics

Recall from [5] that we assume (regardless of mechanical implementation) that the important macroscopic features of a hyper-redundant robot can be captured by a *backbone curve*. A backbone curve parametrization and its associated set of reference frames are collectively called the *backbone reference set*. In this paradigm, inverse kinematics and task planning reduces to the determination of the proper time varying behavior of the backbone reference set [5]. By this approach, global sensor based planning is the determination of the family of backbone curves which describe a hyper-redundant manipulator's path through an unknown environment.

In [8], many techniques are introduced for parametrizing the backbone curve. In this paper, we will assume that the Cartesian position of points on a backbone curve can be parametrized in the form:

$$\vec{x}(s,t) = \int_0^s l(\sigma,t)\ \vec{u}(\sigma,t) d\sigma \qquad (1)$$

where $s \in [0,1]$ is a parameter measuring distance along the backbone curve at time t. The *backbone curve base* is the point $s = 0$. $\vec{x}(s,t)$ is a vector from the backbone curve base to point s. By convention, $\vec{x}(0,t) = 0$. $\vec{u}(s,t)$ is the unit tangent vector to the curve at s. $l(s,t)$ is the length of the curve tangent and assumes the general form:

$$l(s,t) = 1 + \epsilon(s,t) > 0. \qquad (2)$$

$\epsilon(s,t)$ is the *local extensibility* of the manipulator, which expresses how the backbone curve locally expands or contracts relative to a fixed reference state.

The parametrization of Eq. 1 has the following interpretation. The backbone curve is "grown" from the base by propagating the curve forward along the tangent vector, which is varying its direction according to $\vec{u}(s,t)$ and varying its magnitude (or 'growth-rate') according to $l(s,t)$.

In the planar case, the backbone curve is the locus of points:

$$\vec{x}(s,t) = [x_1(s,t),\ x_2(s,t)]^T$$

where

$$x_1(s,t) = \int_0^s l(\sigma,t) \sin\theta(\sigma,t) d\sigma \qquad (3)$$

$$x_2(s,t) = \int_0^s l(\sigma,t) \cos\theta(\sigma,t) d\sigma. \qquad (4)$$

$\theta(s,t)$ is the angle, measured clockwise, which the tangent to the curve at s makes with the x_2-axis at time t. By convention (2.4), $\theta(0) = 0$, and $x_1(0) = x_2(0) = 0$. By comparing equations 1 with equations 3 and 4, it easy to see that $\vec{u}(s,t) = [\sin\theta(s,t),\ \cos\theta(s,t)]^T$ in the planar case. $l(s)$ and $\theta(s)$ are termed "shape functions," as they control they shape of the backbone curve through the forward kinematic relations 3 and 4.

Within the context of this modeling technique, the inverse kinematic problem, or "hyper-redundancy resolution" problem, reduces to the determination of the time varying behavior of backbone curve shape functions that satisfies task requirements. Different hyper-redundancy resolution techniques can be found in [3], [7], [6], [8], [2]. In one approach the backbone curve shape functions are

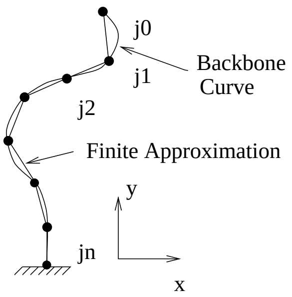

Fig. 4. Backbone Nomenclature

restricted to a "modal form"

$$\theta(s,t) = \sum_{i=1}^{N_\theta} a_i(t)\ \Phi_i(s) \quad l(s,t) = \sum_{i=N_\theta+1}^{N_l} a_i(t)\ \Phi_i(s) \qquad (5)$$

where $\Phi_i(s)$ is a "mode function," and $a_i(t)$ is the associated "modal participation factor." $N = N_\theta + N_l$ is the total number of modes, which must equal or exceeds the number of task constraints. Inverse kinematics reduces to the determination of the modal participation factors.

In [3], motion planning of a hyper-redundant manipulator is reduced to the determination of a family of backbone curves which prescribe the collision-free configurations of the robot as the head moves through an environment. Specifically, [3] suggests an approach where portions of the hyper-redundant manipulator backbone curve are constrained in order to avoid obstacles. A constrained portion of the backbone curve is termed a *tunnel*, and in [3] a tunnel is constructed using full knowledge of the world's geometry. However, Chirikjian and Burdick specify the tunnels by hand, and thus in this work we introduce methods for generating the tunnel segments, using a geometric structure called a roadmap.

A continuous backbone curve inverse kinematic solution is used to determine the actuator displacements of a continuous morphology robot such as one constructed from pneumatic actuator bundles. For discretely segmented morphologies, such as the prototypes described above, the continuous curve solution can be used, via a "fitting" process, to compute the actuator displacements which cause the manipulator to closely approximate the continuous backbone curve model (Figure 4) Examples of such fitting techniques are reviewed in [7] and [8].

3.2 Generalized Voronoi Graph

This work uses a roadmap termed the Generalized Voronoi Graph. In m-dimensions, a *Generalized Voronoi Edge* is the set of points equidistant to m obstacles, such that each point on the edge is closer to the m obstacles than any other obstacle. A GVG edge is one dimensional [12] and GVG edges intersect at *Generalized Voronoi Vertices.* A Generalized Voronoi Vertex is a point equidistant to $m+1$ obstacles such that no other obstacle is closer to the $m+1$ obstacles. The Generalized Voronoi Graph is the collection of GVG edges and vertices.

In the planar case, a GVG edge defined by convex obstacles C_i and C_j is denoted $\mathcal{F}_{ij} = \{x \in R^2 : 0 \leq d_i(x) = d_j(x) \leq d_h(x)\ \forall h\}$, where $d_i(x)$ is the distance between x and C_i. Furthermore, in the plane, a GVG vertex defined by the obstacles C_i, C_j and C_k is denoted, $\mathcal{F}_{ijk} = \{x \in R^2 : 0 \leq d_i(x) = d_j(x) = d_k(x) \leq d_h(x)\ \forall h\}$. In the planar case, the GVG reduces to the Generalized Voronoi Diagram, and both structures are connected [12]. See Figure 5

Another key feature of the GVG is that it can be constructed using solely line of sight information [13]. The GVG edge incremental construction technique, described in [13], is an adaptation of a numerical curve tracing procedure [14]. Assume the robot starts at point on a GVG edge. Using sensor information, it computes the tangent to the graph edge and takes a step in a finite direction along the tangent. This step is sometimes called a "prediction step." Then, on a line orthogonal to the tangent, the robot invokes a correction routine which brings the robot back to the GVG edge. For a finite step size along the tangent, the robot is guaranteed to converge back to the GVG edge during the correction process. Both the prediction step and correction routine can be done in a sensor based manner [13].

Edge tracing continues until one of two conditions are met: (1) the robot encounters a Generalized Voronoi Vertex, or (2) the robot hits a boundary point. A Generalized Voronoi Vertex is sometimes called a *meet point* because that is where three GVG edges meet. At this point, the robot is equidistant to three obstacles. However, due to sensor noise, it is unreasonable to expect the robot to sense this condition. However, meet points can be robustly detected by looking for an abrupt change in the two nearest obstacles. When this occurs, the robot has passed from one GVG edge to another, i.e., it passed by a meet point. Now, the robot notes the approximate location of the meet point and continues tracing a new GVG edge until it either reaches another meet point or a boundary point. A boundary point is where a GVG edge intersects the boundary. At this point, the robot simply turns around and returns to a previous meet point that has unexplored directions. Once all meet points have all directions explored, the robot terminates its exploration process (Figure 5). The completed structure is set of one dimensional paths which simply and usefully reflect the topology of the environment.

3.3 Clipped GVG

We have developed a novel technique for considering the thickness of the robot without computing the configuration space. Consider a disk or cylinder mobile robot with a non-zero diameter. By setting the detection tolerance for boundary points equal to the robot's radius, we effectively construct the GVG of the configuration space without having to compute the configuration space. This technique takes advantage of the GVG's property of being determined from sensor based range information alone. The resulting structure, known as the Clipped Generalized Voronoi Graph or CGVG, automatically factors the robot's width into the exploration procedure. Boundary points are placed at the mouth of narrow channels, "clipping" the GVG prematurely. The CGVG is the subset of the of the GVG which can be reached for an orientable mechanism of given thickness.

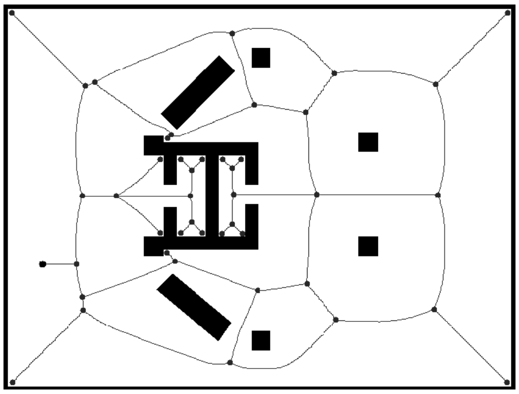

Fig. 5. GVG for a space station cross section

4 Snake Motion Planning

The current motion planning approach marries the backbone curve and GVG approaches, described in the previous section. In the backbone curve paradigm, motion planning is achieved by specifying a continuous series of related backbone curves which bring the robot from an initial configuration to a final configuration. This series, in fact, describes progressively achievable configurations that culminate on a target while avoiding obstacles.

The backbone curve is divided into two components: the free curve and the tunnel curve. The free curve is the portion of the serpentine robot that is "outside" a highly convoluted volume and the tunnel curve is the portion that is "inside." Initially, the serpentine robot entirely fits a free curve. When the serpentine reaches a final configuration inside a highly convoluted environment, then it is largely fit to a tunnel curve.

The GVG of an environment is used to form the tunnel portion of the backbone curve. As the robot moves into the environment, the free curve portion of the backbone diminishes while the tunnel curve grows along the GVG. The result is a continuous parameterization for backbone curves which specify a path that the serpentine robot follows to achieve a goal configuration.

Typically, a serpentine robot is segmented into kinematically sufficient "bays." This means that each bay can arbitrarily position and orient its end-plate with respect to its base. A fitting procedure is used to place and orient each bay along the entire backbone.

Unfortunately, the initial guess of the backbone curve may contain some configurations to which the mechanism cannot be fitted. Joint limits and finite bay length prevent the serpentine from following the backbone exactly. Although the backbone curve is guaranteed to be a collision free path, a discrete segmented approximation of the curve may intersect obstacles. In this case the backbone curve must be continuously deformed to satisfy joint limits and avoid obstacles. After a proper deformation process is completed, the resulting set of backbone curves provides a path for the serpentine robot to reach a goal configuration. Otherwise, the procedure can determine with certainty that no such path exists. A deformation procedure that will satisfy all joint limits while optimizing the backbone curve for curvature and length is a current topic of theoretical research.

5 Simulation

We have developed a planar simulator that will construct the GVG for a highly convoluted environment, then compute the backbone curves and serpentine configurations (Figure 6).

Initially, the entire GVG is generated. This is akin to exploring an unknown environment. The GVG is incrementally constructed using a numerical curve tracing technique as described above. We simulate the sensor range information with a function that computes the absolute distance to convex polygonal and circular obstacles (although the GVG applies to obstacles of any geometry). The code runs on a Sun Workstation and can compute the GVG for an unstructured two-dimensional environment in typically fewer than five seconds.

As with other computational geometry problems, the construction of the GVG is complicated by the presence of symmetries in the environment, such as those in Figure 5. Meet points typically occur at triple equidistance, where there are three local minima in the robot's visible distance function. Hyper-symmetric environments contain degenerate points that are equidistant to four or more obstacles (e.g. at the crossroads of two perpendicular corridors). The simulator has been developed for robustness in generating the GVG for such n-way meet points. This involves the definition of new GVG branches as running between angularly sorted pairs of sensor minima.

There are two basic modes in which path planning can be achieved: (1) exploration and (2) start-goal motion planning. Exploration either generates the full GVG using line of sight information or uses an existing GVG to re-perform a depth-first search of the environment.

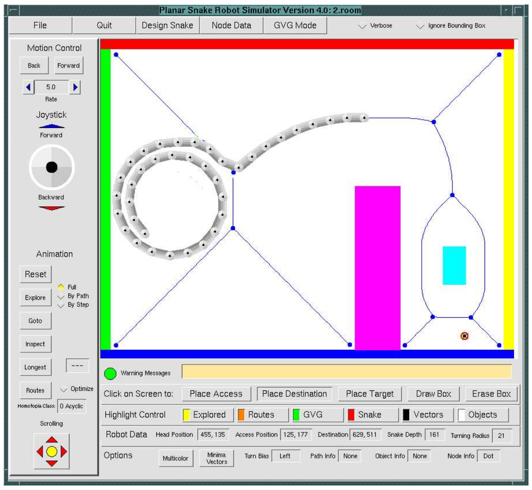

Fig. 6. Snake Robot Simulator

There are two sub-modes for start-goal motion planning: (i) goal location is known (maze searching) and (ii) goal location is not known (inspection). The known goal search uses a best-first search heuristic to reach a goal with known coordinates. Each time the robot encounters a meet point, it choses a new edge to explore which locally minimizes the robot's distance to the goal. If the coordinates of the goal location are not known, then the simulator explores the GVG until the goal is within line of sight. This is akin to searching for trapped survivors in search-and-rescue operations or inspecting a complex structure for defects. We assume that the robot can identify the goal at a distance by sight.

We use the GVG's property of *departability* to reach an arbitrary goal location in the free space. Recall that *departability* guarantees that an obstacle free straight line exists to every point in the environment from the GVG roadmap. When the distance to the goal (known from its absolute coordinates) becomes smaller than the distance to any other object, the robot may assume it is safe to depart the graph by defining a new roadmap segment straight from its current location to the goal.

Motion planning for a serpentine robot is achieved via a follow-the-leader approach. The head of the snake moves along the GVG, while the rest of body follows. We assume that if the head in not already at a boundary point, that it is possible to advance the snake such that the head moves an increment δs farther along the graph. The result of the fitting procedure is later analyzed to check if this assumption is correct, and if so, we update the robot.

After assigning a new head position, the fitting rou-

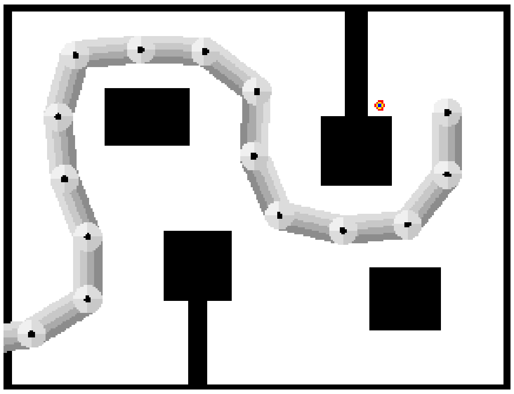

Fig. 7. Simulated Serpentine Robot

tine proceeds to step back along the GVG, placing joint centers a fixed distance L from the center of the preceding joint. Joint indices start at zero on the head, and count upwards. The the fitting procedure is performed for the entire backbone curve which includes the tunnel curve and the free curve. The free curve is generally a spiral storage shape which unwinds as the serpentine enters the workspace. The computing time to fit the snake is a linear function of the total number of links in the mechanism.

The GVG naturally contains a number of sharp angle discontinuities between separate segments arriving at a meet point. The backbone for a given path will have curvature discontinuities at meet points where it switches from one GVG segment to another. Because the snake joint placement represents a finite approximation of the GVG backbone, such corners are cut. This in general smoothes out sharp discontinuities in the backbone and allows the snake to maintain small joint angles (Figure 8). We have found that even with highly discontinuous backbone curves, joint limits are rarely violated. Since we rarely have to invoke a joint limit handler, the computing time to find a complete set of backbone curves for every time step in a task will not be much greater than the current computing time.

Sharp discontinuities in the shape of the backbone also lead to irregularities in the snake's time evolution as it advances into the environment. As the snake moves, different corner cutting arrangements around discontinuities cause a continuous forward motion at the head to become discontinuous at the tail. The tail of the snake may even be commanded move backwards in order to advance the head. This occurs when the fitting procedure cuts a corner that was not cut in the previous arrangement, resulting in less overall length required to reach the desired head position.

We have also written a routine to compute the farthest point on the graph from an arbitrary access point. By knowing the farthest point, we can prevent the snake

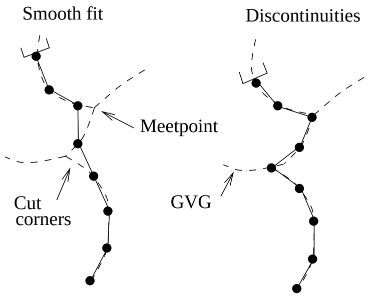

Fig. 8. Effect of fitting procedure on discontinuous backbone

from wrapping unnecessarily around the environment in a depth-first search. This problem occurs for highly cyclic GVG's, when a depth-first search performs most of its activity with the snake wrapped around distant obstacles while exploring segments more easily accessed from other directions. The serpentine can be instructed never to search deeper into the environment than the distance to the farthest point, since a shorter less convoluted route to any such end effector location is known to exist.

A snake robot designer, included for testing purposes, can be used to design a snake for a given environment. The essential parameters used to size a snake are the number of bays, bay length, bay width, and maximum joint angle.

6 Conclusion and Future Work

Using the Generalized Voronoi Graph for serpentine robot motion planning has several advantages. First, it can be performed in a sensor based way, without a priori knowledge of the environment, which is more appropriate for the complex environments where serpentine robots will be most beneficial. Secondly, using a follow-the-leader approach to define backbone curves through the environment, the computing cost associated with a highly redundant manipulator can be greatly reduced. Computational efficiency allows for real time control without full knowledge of the environment. Savings is achieved by not having to perform the manipulator inverse kinematics, not having to compute to configuration space, and by reducing an m dimensional environment to a one dimensional roadmap search space.

An important aspect of this technique is that is can be generalized to three dimensional environments without a significant increase in computation. The fitting procedure for a series of discreet links along a continuous backbone curve is no more computationally expensive than in the two-dimensional case. The time required to compute and update a snake configuration in a three-dimensional environment is of the same order of magnitude as in the two-dimensional case. For the simulation, the computing time required to update a three-dimensional display overwhelms the time used to configure the snake.

Using the GVG all-ready maximizes safety for a given route through the environment by keeping the snake joints as far away from the walls as possible. Optimizing safety is desirable in many real applications such as navigating under the rubble of a collapsed building.

Currently our simulation assumes no angular joint limits on the device. Various methods for handling these limits exist, including local and global solutions. One method would compute the joint position list as an initial guess and search that list for limit violations. regions with curvature above the rated maximum could be locally deformed into a section of maximum curvature with continuity boundary constrains with the rest of the curve.

Although using the GVG ensures that the backbone curve is placed midway between all obstacles and never intersects with corners, a real snake configuration using links with a high length to width ratio will fit to a coarse approximation of the backbone. This leads to the possibility of the coarse discrete approximation cutting corners. This requires a means to detect obstacle collisions in the discrete approximation and a process for deforming the initial backbone away from the problem area, similar to the method reported in [18].

A proper solution to the joint limits problem involves the global optimization of the backbone curve with an upper limit on the backbone's curvature. This method is considered superior to the problem of local joint limit handling. We are currently developing a technique using the calculus of variations to optimize the the backbone curve for cost functions involving length, curvature, safety, and energy. We plan to use the GVG as a first approximation to the backbone curve, then iteratively deform this seed curve while considering numerous obstacle constraints, and a limited maximum curvature. Finally, experiments on the eleven degree of freedom JPL manipulator are underway.

References

[1] J.W. Burdick, J. Radford, and G.S. Chirikjian. A 'Sidewinding' Locomotion Gait for Hyper-Redundant Robots. In *IEEE Int. Conf. on Robotics and Automation*, Atlanta, GA, May 1993.

[2] G.S. Chirikjian. *Theory and Applications of Hyper-Redundant Robotic Manipulators.* PhD thesis, California Institute of Technology, Pasadena, CA, 1992.

[3] G.S. Chirikjian and J.W. Burdick. An Obstacle Avoidance Algorithm for Hyper-Redundant Manipulators. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 625–631, Cincinnati, Ohio, May 14-17 1990.

[4] G.S. Chirikjian and J.W Burdick. Applications of Hyper-Redundant Manipulators for Space Robotics and Automation. In *Proc. Int. Symposium on Artificial Intelligence and Robotics Applications to Space*, Kobe, Japan, November 1990.

[5] G.S. Chirikjian and J.W. Burdick. Kinematics of Hyper-Redundant Manipulators. In *Proc. ASME Mechanisms Conference*, pages 391–396, Chicago, IL, Sept. 16-19 1990.

[6] G.S. Chirikjian and J.W. Burdick. Kinematics of Hyper-Redundant Locomotion with Applications to Grasping. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Nice, France, May 10-15 1991.

[7] G.S. Chirikjian and J.W. Burdick. Parallel Formulation of the Inverse Kinematics of Modular Hyper-Redundant Manipulators. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 708–713, Sacramento, CA, April 1991.

[8] G.S. Chirikjian and J.W. Burdick. A Modal Approach to Hyper-Redundant Manipulator Kinematics. *IEEE Trans. on Robotics and Automation*, June 1994.

[9] G.S. Chirikjian and J.W. Burdick. Kinematically Optimal Hyper-Redundant Manipulator Configurations. *IEEE Trans. on Automation and Robotics*, December 1995.

[10] G.S. Chirikjian and J.W. Burdick. The Kinematics of Hyper-Redundant Robotic Locomotion. *IEEE Trans. on Automation and Robotics*, December 1995.

[11] H. Choset and J.W. Burdick. Sensor Based Planning and Nonsmooth Analysis. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 3034–3041, San Diego, CA, 1994.

[12] H. Choset and J.W. Burdick. Sensor Based Planning, Part I: The Generalized Voronoi Graph. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Nagoya, Japan, 1995.

[13] H. Choset and J.W. Burdick. Sensor Based Planning, Part II: Incremental Construction of the Generalized Voronoi Graph. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Nagoya, Japan, 1995.

[14] H.B. Keller. *Lectures on Numerical Methods in Bifurcation Problems.* Tata Institute of Fundamental Research, Bombay, India, 1987.

[15] J.C. Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, Boston, MA, 1991.

[16] C. Ó'Dúnlaing and C.K. Yap. A "Retraction" Method for Planning the Motion of a Disc. *Algorithmica*, 6:104–111, 1985.

[17] D. Reznik and V. Lumelsky. Motion Planning with Uncertainty for Highly Redundant Kinematic Structures I. 'Free Snake' Motion. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Raleigh, N.C., 1992.

[18] N. Takanashi, H. Choset, and J.W. Burdick. Local sensor based planning for hyper-redundant manipulator. In *Proc. of IEEE Int. Conf. on Intelligent Robots and Systems*, Yokohama, Japan, July 1993.