PCG: A Foothold Selection Algorithm for Spider Robot Locomotion in 2D Tunnels

Amir Shapiro and Elon Rimon
Technion, Israel Institute of Technology
amirs@tx.technion.ac.il, elon@robby.technion.ac.il

Abstract *This paper presents an algorithm, called* PCG, *for planning the foothold positions of spider-like robots in planar tunnels bounded by piecewise linear walls. The paper focuses on* 3*-limb robots, but the algorithm generalizes to robots with a higher number of limbs. The input to the PCG algorithm is a description of a tunnel having an arbitrary piecewise linear geometry, a lower bound on the amount of friction at the contacts, as well as start and target foothold positions. Using efficient convex programming techniques, the algorithm approximates the possible foothold positions as a collection of cubes in contact c-space. A graph structure induced by the cubes has the property that its edges represent feasible motion between neighboring sets of* 3*-limb postures. This motion is realized by lifting one limb while the other two limbs brace the robot against the tunnel walls. A shortest-path search along the graph yields a* 3-2-3 *gait pattern that moves the robot from start to target using a minimum number of foothold exchanges. Simulation results demonstrate the PCG algorithm in a tunnel environment.*

1 Introduction

Many robotic tasks are suited for legged robots that interact with the environment in order to achieve stable locomotion. For example, surveillance of collapsed structures for survivors [13], inspection and testing of complex pipe systems [9], and maintenance of hazardous structures such as nuclear reactors [11]. Our ultimate goal is to develop spider-like mechanisms that navigate quasistatically in such complex environments. A *spider-like robot* consists of k articulated limbs attached to a central body, such that each limb ends with a *footpad* (Figure 1). This paper presents an algorithm called PCG (short for Partitioned Cubes Gaiting), for planning the foothold positions of spider-like robots in planar tunnel environments.

In our setup, the robot moves by exerting forces on tunnel walls which are mounted on a horizontal plane. The robot is supported against gravity by frictionless contacts mounted under the mechanism (Figure 1). In general, a spider-like robot must have at least *three* limbs in order to move quasistatically in a planar tunnel[1]. At every instant the spider braces against the tunnel walls in static equilibrium using two or three limbs. During a 2-limb posture the spider moves its free limb to the next foothold position. During a 3-limb posture the spider changes its internal geometry in preparation for the next limb lifting. The PCG algorithm is presented in the context of such 3-limb robots, but we also discuss the generalization of the algorithm to higher number of limbs.

We make the following assumptions. First, we assume piecewise linear tunnel walls with known geometry. Second, the entire tunnel lies in a horizontal plane so that gravity is excluded. Third, each limb can only push against the environment, using its footpad. Fourth, each footpad contacts the tunnel walls via a frictional point contact, with a known lower bound on the coefficient of friction. The i^{th} foothold position is parametrized by $s_i \in [0, L]$, where L is the total length of the tunnel walls. The footholds of the entire k-limb mechanism are parametrized by *contact c-space*, $(s_1, \ldots, s_k) \in [0, L]^k$ (Figure 2). Last, we lump the kinematic structure of the robot into a single parameter called the *robot radius.* This parameter, denoted R, is the length of a fully stretched limb. The algorithm uses R to ensure that the selected footholds can be reached from the robot's central base.

Relationship to prior work. The use of contact c-space is common in the grasp planning literature. In particular, Nguyen [8] and Ponce et al. [10] introduced the notion of *contact independent regions.* Given a k-finger grasp of a planar object, a contact independent region is a k-dimensional cube in contact c-space aligned with the coordinate axes. This cube represents k segments along the object's boundary, such that any placement of the k contacts inside these segments generates an equilibrium grasp. We use a similar notion in our representation of the feasible footholds as cubes in contact c-space. Each cube represents three segments along the tunnel walls, such that any placement of three footpads inside these seg-

[1] In quasistatic motion inertial effects due to moving parts of the robot are kept small relative to the forces of interaction between the robot and the environment.

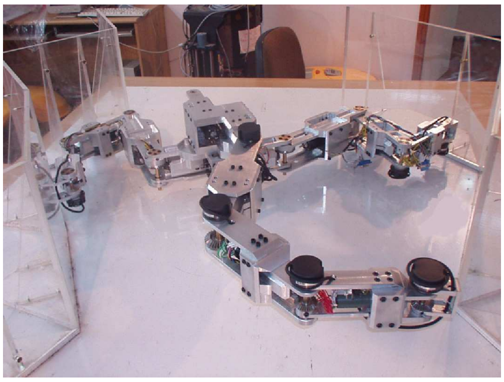

Figure 1: Top view of a 3-limb spider robot moving in a planar tunnel environment.

ments results in a feasible 3-limb equilibrium posture. Other relevant grasping papers discuss finger gaiting. Hong et al. [3] describe 3 and 4-finger gaits for planar objects. However, they assume that once an object is grasped, the fingers may not change their order along the object's boundary. In contrast, we impose no restriction on the order of the footpads along the tunnel walls. Goodwine et al. [15] investigate the stratification of the full configuration space associated with finger gaiting. While their approach is justifiable for the design of feedback control laws, motion planning can be carried out in lower dimensional spaces such as contact c-space.

In the multi-legged locomotion literature, Boissonnat et al. [1] discuss a motion planning algorithm for multi-legged robots that move in a gravitational field over a flat terrain. Much like our approach, they lump the kinematic structure of the robot into a reachability radius, and compute a sequence of stable stances from start to target. Our work differs from the work of Boissonnat et al. in two fundamental ways. First, we consider motions where the robot stably braces against tunnel walls rather than maintaining stable stances against gravity. Second, they allow the legs to contact only discrete point sites, while we allow arbitrary footpad placement along the tunnel walls. Other papers that consider motion planning for multi-legged robots are [2, 4, 6, 7, 14].

This paper focuses on the portion of the PCG algorithm that plans a sequence of footholds in contact c-space. In Section 2 we characterize the feasible 3-limb postures in contact c-space. These postures must be reachable, form stable equilibria, and satisfy a condition that allows their inclusion in a 3-2-3 gait pattern. In Section 3 we establish a key result, that the feasible 3-limb postures are a union of convex sets in contact c-space. It is also shown in this section that the approximation of a convex set by p maximal cubes is a convex optimization problem. In Section 4 we describe the PCG algorithm. The algorithm approximates the convex sets by contact independent cubes, then searches a graph induced by the cubes for the shortest sequence of footholds from start to target. In Section 5 we run the PCG algorithm on a simulated tunnel environment. Finally, in the concluding section we discuss the generalization of the algorithm to higher number of limbs.

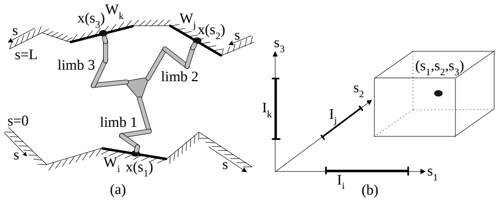

Figure 2: (a) A 3-limb robot in a planar tunnel, and (b) the parametrization of its contact c-space.

2 The Feasible 3-Limb Postures

In this section we characterize the feasible 3-limb postures as inequality constraints in contact c-space. The feasible 3-limb postures must form stable equilibria, be reachable, and satisfy the following *gait feasibility* condition. This condition requires that a 3-limb posture will contain two distinct 2-limb postures—one for entering the 3-limb posture by establishing a new foothold, and one for leaving the 3-limb posture by releasing some other foothold.

Next we introduce notation that would allow us to write the above conditions as inequalities in contact c-space. Let $W_1, \ldots, W_n$ denote the tunnel walls, and let $I_1, \ldots, I_n$ be the partition of $[0, L]$ into intervals that parametrize the individual walls (Figure 2). Let t_i and n_i denote the unit tangent and unit normal to the i^{th} wall. Points along W_i are given by $\boldsymbol{x}(s) = x_i + st_i$, where x_i is the initial vertex of W_i and $s \in I_i$. Given a contact force $\boldsymbol{f}_i$, $f_i^t = \boldsymbol{f}_i \cdot t_i$ and $f_i^n = \boldsymbol{f}_i \cdot n_i$ are the tangent and normal components of $\boldsymbol{f}_i$. The friction cone at a contact along the i^{th} wall, denoted FC_i, is given by $FC_i = \{\boldsymbol{f}_i : f_i^n \geq 0 \text{ and } -\mu f_i^n \leq f_i^t \leq \mu f_i^n\}$, where μ is the coefficient of friction.

Gait feasibility requires that a 3-limb will contain two distinct 2-limb postures. As a preparation, we review the conditions for equilibrium and stability of 2-limb postures. A 2-limb mechanism forms an equilibrium posture if the line segment connecting the two contacts lies inside the two friction cones [8]. As a stability criterion we use the notion of *force closure*—a posture where the mechanism can resist any perturbing wrench by suitable adjustment of its contact forces. In general, a planar equilibrium posture is force closure and hence stable if the contact forces of the unperturbed posture lie in the interior

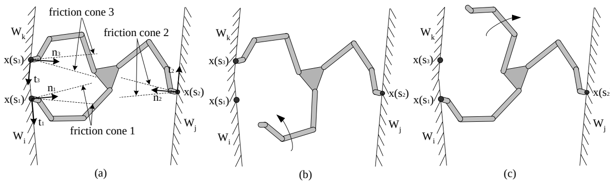

Figure 3: (a) A gait feasible 3-limb posture, (b)-(c) contains two distinct 2-limb postures.

of the respective friction cones.

Let two limbs with indices l and m contact the tunnel walls W_i and W_j. Then for a 2-limb stable equilibrium, the vector $x(s_m)-x(s_l)$ must lie in the interior of FC_i, while $-(x(s_m)-x(s_l))$ must lie in the interior of FC_j. This condition defines a set in the (s_l, s_m) plane, denoted $\mathcal{E}_{ij}^{lm}$, given by

$$\begin{aligned}\mathcal{E}_{ij}^{lm} \;&= \{(s_l, s_m) \in I_i \times I_j : \\ &\big|\big(x(s_m)-x(s_l)\big)\cdot \boldsymbol{t}_i\big| < \mu\big(x(s_m)-x(s_l)\big)\cdot \boldsymbol{n}_i, \\ &\big|\big(x(s_l)-x(s_m)\big)\cdot \boldsymbol{t}_j\big| < \mu\big(x(s_l)-x(s_m)\big)\cdot \boldsymbol{n}_j\}.\end{aligned}$$

It is important to note that the inequalities describing $\mathcal{E}_{ij}^{lm}$ are linear in s_l and s_m. Hence $\mathcal{E}_{ij}^{lm}$ is a *convex polygon* in the (s_l, s_m) plane. When $\mathcal{E}_{ij}^{lm}$ is considered as a subset of the contact c-space of a 3-limb mechanism, it becomes a three-dimensional prism orthogonal to the (s_l, s_m) plane. The prism is denoted with an $\times$ for the limb that does not participate in the 2-limb posture. The 2-limb equilibrium set $\mathcal{E}_{ij}^{12}$ thus becomes the prism $\mathcal{P}_{ij\times}$, the sets $\mathcal{E}_{ij}^{13}$ becomes $\mathcal{P}_{i\times j}$, and $\mathcal{E}_{ij}^{23}$ becomes $\mathcal{P}_{\times ij}$.

Reachability constraint of 3-limb postures. A 3-limb posture is reachable when its footholds lie within the robot's radius R. For each wall triplet W_i, W_j, W_k the reachability constraint is given by

$$\begin{aligned}&\mathcal{R}_{ijk} = \{(s_1, s_2, s_3) \in I_i \times I_j \times I_k : \exists c \in I\!R^2 \\ &\max\{\|x(s_1)-c\|, \|x(s_2)-c\|, \|x(s_3)-c\|\} \leq R\},\end{aligned} \tag{1}$$

The point c appearing in (1) can be interpreted as the center of a disc containing the three foothold positions, such that the disc radius is bounded by R. As discussed below, the elimination of the existential quantifier in (1) results in a set which is bounded by quadratic surfaces in contact c-space.

Gait feasibility of 3-limb postures. A 3-limb posture is gait feasible if it contains two distinct 2-limb equilibrium postures (Figure 3). Let us write this constraint in a cell $I_i \times I_j \times I_k$ of contact c-space. This cell corresponds to contact with the walls W_i, W_j, W_k, and gait feasibility is satisfied by intersection of pairs of 2-limb prisms associated with the three walls. There are three such pairs—$(\mathcal{P}_{ij\times}, \mathcal{P}_{i\times k})$, $(\mathcal{P}_{ij\times}, \mathcal{P}_{\times jk})$, and $(\mathcal{P}_{\times jk}, \mathcal{P}_{i\times k})$—and the resulting set of feasible 3-limb postures in the cell, denoted $\mathcal{F}_{ijk}$, is given by

$$\begin{aligned}\mathcal{F}_{ijk} = \;& \big(\mathcal{P}_{ij\times} \cap \mathcal{P}_{i\times k} \cap \mathcal{R}_{ijk}\big) \cup \big(\mathcal{P}_{ij\times} \cap \mathcal{P}_{\times jk} \cap \mathcal{R}_{ijk}\big) \\ & \cup \big(\mathcal{P}_{\times jk} \cap \mathcal{P}_{i\times k} \cap \mathcal{R}_{ijk}\big).\end{aligned} \tag{2}$$

Note that the same three walls appear in *six* cells in contact c-space, each corresponding to a specific assignment of the limbs to the three walls. The entire collection of feasible 3-limb postures is the union of all such sets over all ordered wall triplets. We end with the following assertion [12]. *It is always possible to transfer forces between two 2-limb postures contained in a feasible 3-limb posture, while the mechanism is kept in static equilibrium with three fixed footholds.*

3 Convexity of Feasible 3-Limb Postures

In this section we establish two convexity results that will be used by the PCG algorithm. First we establish that the feasible 3-limb postures are a union of convex sets in contact c-space. Then we show that the approximation of a convex set by p maximal cubes is a convex optimization problem.

3.1 Convexity of the feasible postures

The set $\mathcal{F}_{ijk}$ of feasible 3-limb postures is specified in (2) as a union of three sets, each corresponding to a particular pair of 2-limb postures. The following lemma asserts that each of these sets is convex in contact c-space.

Lemma 3.1. *In each cell $I_i \times I_j \times I_k$ of contact c-space, the set $\mathcal{F}_{ijk}$ of feasible 3-limb postures is a union of three convex sets.*

Proof: Consider the set $\mathcal{P}_{ij\times} \cap \mathcal{P}_{\times jk} \cap \mathcal{R}_{ijk}$ in (2). The prisms $\mathcal{P}_{ij\times}$ and $\mathcal{P}_{\times jk}$ are defined by intersection of linear inequalities, and are therefore convex polytopes in contact c-space. Next consider the set $\mathcal{R}_{ijk}$. The existential quantifier in (1) acts on a set, denoted $\bar{\mathcal{R}}_{ijk}$, which is defined in the five-dimensional space (s_1, s_2, s_3, c): $\bar{\mathcal{R}}_{ijk} = \{(s_1, s_2, s_3, c) \in I_i \times I_j \times I_k \times I\!R^2 : \max\{\|x(s_1)-c\|, \|x(s_2)-c\|, \|x(s_3)-c\|\} \leq R\}$. The norm function $\|x-c\|$ is convex in (x, c) space, and each $x(s_i)$ is linear in s_i. Hence the functions $\|x(s_i)-c\|$ are convex in (s_1, s_2, s_3, c) space. The pointwise maximum of convex functions is a convex function. Hence $\bar{\mathcal{R}}_{ijk}$ is convex in (s_1, s_2, s_3, c) space. But $\mathcal{R}_{ijk}$ is the projection of $\bar{\mathcal{R}}_{ijk}$ onto contact c-space. Since projection preserves convexity, $\mathcal{R}_{ijk}$ is convex. Finally, the intersection of convex sets is convex, hence $\mathcal{P}_{ij\times} \cap \mathcal{P}_{\times jk} \cap \mathcal{R}_{ijk}$ is convex. □

To summarize, the set $\mathcal{F}_{ijk}$ is the union of three convex sets, each bounded by planar surfaces associated with the 2-limb prisms, and quadratic surfaces associated with the reachability constraint [12].

3.2 Convexity of Cube Approximation

We now discuss the approximation of the convex sets comprising $\mathcal{F}_{ijk}$ by maximal cubes. Consider the approximation of a three-dimensional convex set $\mathcal{S}$ by p cubes, where the cubes have arbitrary center and dimensions. We assume as input a desired relative configuration for the p cubes, where a *relative configuration* is a specification of an adjacency relation between the cubes in terms of a set of separating planes, such that no two cubes can possibly intersect. Each of the separating planes is defined in terms of the relative position of two cubes, and does not restrict the absolute position of the two cubes. The i^{th} cube is parametrized by its center $c_i \in I\!R^3$, and its dimensions along the coordinate axes, $h_i \in I\!R^3$. The optimization therefore takes place in the $6p$-dimensional space whose coordinates are $(c_1, h_1, \ldots, c_p, h_p)$. Our objective is to maximize the total volume of the cubes. However, the sum of the cubes' volumes is not a convex function of the optimization variables. Rather, we use a normalized total volume function given by[2]

$$\phi(c_1, h_1, \ldots, c_p, h_p) = \sum_{i=1}^{p} (h_{i1} h_{i2} h_{i3})^{\frac{1}{3}}.$$

Next we list the constraints involved in the cube approximation problem. First we have the requirements that the cubes' dimensions be non-negative, $h_{ij} \geq 0$, and that their centers lie inside contact c-space, $0 \leq c_{ij} \leq L$ $(i = 1, \ldots, p, j = 1, 2, 3)$. Second, the relative configuration of the cubes is specified by a list of separating planes, each involving the center and dimensions of two cubes separated by the plane. Last, we must ensure that the cubes lie inside the convex set $\mathcal{S}$. The following proposition asserts that the maximization of ϕ over p cubes contained in $\mathcal{S}$ is a convex optimization problem.

Proposition 3.2 ([12]). *The maximization of* $\phi = \sum_{i=1}^{p} (h_{i1} h_{i2} h_{i3})^{\frac{1}{3}}$ *over* p *cubes contained in a convex set* $\mathcal{S}$ *and satisfying a relative-configuration specification is a convex optimization problem.*

It is worth mentioning that convex optimization algorithms, for instance the ellipsoid algorithm used in our implementation, generate an ϵ-accurate solution in $O(m^2 l \log(1/\epsilon))$ time, where m is the number of optimization variables and l the number of steps required to evaluate the constraints.

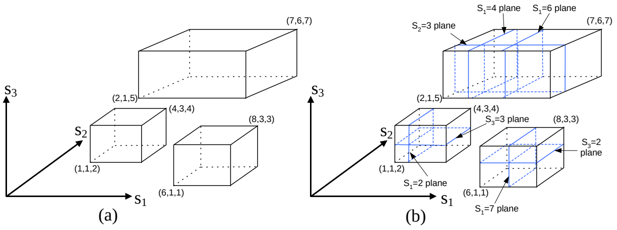

Figure 4: (a) Three cubes in contact c-space, and (b) their mutual partition into sub-cubes along the separating planes.

4 The PCG Algorithm

We begin with an overview of the algorithm. The set of feasible 3-limb postures in each contact c-space cell, $\mathcal{F}_{ijk}$, is a union of three convex sets. However, usually each cell contains at most one convex set, and we describe the algorithm under the assumption of a single convex set per cell. The algorithm first approximates each of the convex sets by p maximal cubes. The number of cubes and their relative configuration are user-specified inputs whose selection is discussed below. In order to describe the next stage of the algorithm we introduce the notion of cube orientation. A maximal cube parametrizes a set of feasible 3-limb postures, each containing two distinct 2-limb postures. The two 2-limb postures necessarily share a limb in common (Figure 3). However, this common limb *cannot be lifted,* since its lifting would destroy both 2-limb postures. By construction, all the 3-limb postures parametrized by a given maximal cube have the same common limb. Thus we associate with each maximal cube an *orientation vector,* which is aligned with the s_i-axis of the limb that cannot be lifted from the 3-limb postures parametrized by the cube.

In the second stage the algorithm partitions the maximal cubes as follows. The algorithm constructs an arrangement of all the separating planes of the cubes, where each separating plane contains one of the cubes' faces. Using this arrangement, the algorithm partitions the cubes as illustrated in Figure 4. The figure shows three cubes and their mutual partition along the separating planes into sub-cubes. During the partition process, each sub-cube inherits the orientation vector of its parent cube. The resulting sub-cubes have disjoint interiors, and they satisfy the following projection property. Any two sub-cubes either have precisely the same projection on one of the coordinate planes, or their projection on all three coordinate planes are disjoint. If two sub-cubes share a projection they are called *compatible*, and the s_i-axis aligned with the direction of projection is called the *direction of compatibility.*

In the third stage the algorithm constructs a graph called the *sub-cube graph.* The nodes of the

[2] This function was suggested to us by Prof. A. Nemirovsky.

graph are center points of the sub-cubes. The edges of the graph are assigned unit weights. Each edge connects compatible sub-cubes whose direction of compatibility is orthogonal to the orientation vector of the two sub-cubes. The meaning meaning of the orthogonality condition is discussed below. Finally, the start and target 3-limb postures, denoted S and T, are added as special nodes to the sub-cube graph. The construction of edges from S and T to the other nodes of the graph is described below.

In the last stage, the algorithm searches the sub-cube graph for the shortest path from S to T. The shortest path on the graph minimizes the number of foothold exchanges along the path from S to T. However, this minimality is relative to the cube approximation obtained in the first stage of the algorithm. A formal description of the algorithm follows.

PCG Algorithm:
Input: Geometrical description of an n-wall tunnel. A value for the coefficient of friction. Start and target 3-limb postures S and T. A value for the number of cubes p and their relative configuration.

1. Cube approximation:
   1.1 Determine which cells $I_i \times I_j \times I_k$ contain a non-empty set $\mathcal{F}_{ijk}$ of feasible 3-limb postures.
   1.2 Approximate each non-empty set $\mathcal{F}_{ijk}$ by p maximal cubes. Assign an orientation vector to each maximal cube.
2. Cube partition:
   2.1 Construct an arrangement of the separating planes of all maximal cubes.
   2.2 Subdivide each maximal cube into sub-cubes along the separating planes. Assign to each sub-cube the orientation vector of its parent cube.
3. Graph construction:
   3.1 Define a *sub-cube graph* as described above.
   3.2 Define S and T as special nodes and connect them to the graph as described below.
   3.3 Assign unit weight to all edges.
4. Graph search:
   Search for the shortest path along the sub-cube graph from S to T.

Let us discuss the meaning of the edges in the sub-cube graph. An edge represents lifting and replacement of a particular limb. The lifting of a limb must leave the robot with a stable 2-limb posture. The orientation vector of a sub-cube describes which limb my not be lifted from the 3-limb postures parametrized by the sub-cube. Hence all edges emanating from a node must be *orthogonal* to the orientation vector of the sub-cube associated with the node. Moreover, all edges of the sub-cube graph are *straight lines parallel to the s_i-axes in contact c-space* (Figure 6). Another aspect of the edges is reachability—motion of a limb between any two sub-cubes connected by an edge can always be executed such that reachability is maintained throughout the limb's motion [12].

Next consider the construction of edges from S and T to the other nodes of the sub-cube graph. Let S and T be feasible 3-limb postures with their own orientation vector. For S and T, compatibility with a sub-cube means that the projection of the sub-cube on one of the coordinate planes contains the corresponding projection of the node. Having defined orientation and compatibility for S and T, the edges connecting these nodes to the other nodes of the graph are constructed by the rule specified in step 3.1 of the algorithm. A second technical issue is the selection of a relative configuration for the p cubes. We specify in each cell a relative configuration that separates the p cubes along a coordinate axis which is *orthogonal* to the cell's orientation vector. Adjacent maximal cubes consequently overlap along the cell's allowed directions of motion, thereby preserving the connectivity of the set of feasible 3-limb postures in the cell.

Let us discuss some notable features of the algorithm. First, the uniform weight assignment reflects our desire to minimize the number of foothold exchanges along the path. Second, the algorithm treats the motion of a limb between walls and along a single wall in a uniform manner. Last, the size of the sub-cube graph increases with p. However, if an edge exists in the graph for low values of p, it would persist in the graph for larger values of p. Consequently, *the path from start to target only becomes shorter as p increases.* The computational complexity of the algorithm is analyzed in Ref. [12]. Under the reasonable assumption that the spider robot can reach from any given position only a small number of walls which is bounded by a constant, the algorithm runs in $O(np^6 \log(np))$ time, where n is the number of tunnel walls and p is the number of cubes per cell in contact c-space.

5 Simulation Results

In this section we run the PCG algorithm in the tunnel depicted in Figure 7. The tunnel consists of six walls whose lengths in cm are marked in the figure. The robot reachability radius is $R = 60$ cm, and the coefficient of friction is $\mu = 0.5$. Note that this simple tunnel already contains significant geometric features: the two walls at the bottom form a closing cone, the tunnel next turns leftward and becomes two parallel walls, and finally the two walls at the top form an opening cone. These geometric features are significant, since *the robot must use friction effects to*

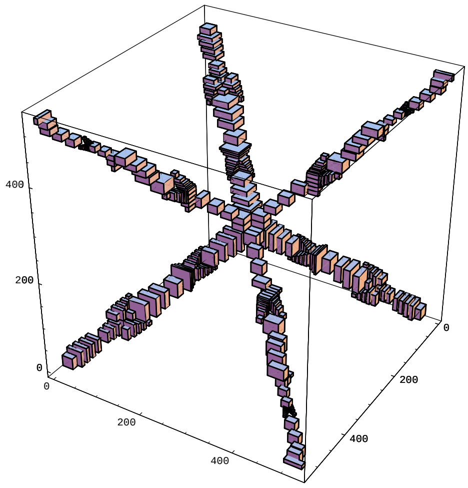

Figure 5: The collection of 270 maximal cubes approximating the feasible 3-limb postures.

traverse such features.

Let us now discuss the computation of the feasible 3-limb postures in contact c-space. The collection of feasible 3-limb postures has a *six-fold symmetry* consisting of six symmetric "arms:" every non-empty cell represents an assignment of the three limbs to a triplet of walls, and there are six permutations of the three limbs on the triplet of walls. The arms are roughly aligned with the diagonals of contact c-space, for the following reason. The coordinate projection of each arm covers the entire length of the tunnel. Each arm can therefore be visualized as "dragging" the 3-limb mechanism as a single rigid body along the entire length of the tunnel. There are nine non-empty cells in each arm, giving a total of 54 non-empty cells in the entire contact c-space.

Next consider the cube approximation of the feasible 3-limb postures. We use $p = 5$ cubes per cell and compute the maximal cubes using the ellipsoid algorithm. The $p = 5$ value preserves the connectivity of the set of feasible 3-limb postures, while still being sufficiently low to allow reasonable execution time. The result of running the ellipsoid algorithm on the non-empty cells of contact c-space appear in Figure 5. Since there are 54 non-empty cells, the resulting cube approximation of contact c-space contains $5 \cdot 54 = 270$ maximal cubes. The algorithm next partitions each of the maximal cubes along the separating planes of the other maximal cubes. The partitioning of the maximal cubes generated 28,299 sub-cubes in each of the six arms of contact c-space (the resulting sub-cubes are not shown).

The algorithm next constructs the sub-cube graph, and searches the graph for the shortest path from the start to target postures. The result of computing the shortest path is shown in Figure 6. Each segment in the figure is an edge of the sub-cube graph that represents one limb lifting and re-placement. Figure 7 shows the same path in physical space, where each foothold is marked by its index in the sequence of steps taken by the robot. Let us denote the sequence of 3-limb postures by (i_1, i_2, i_3), where i_j is the foothold position of limb j at the i^{th} posture. Then the path computed by the algorithm consists of the 3-limb postures: $S = (1, 2, 3) \rightarrow (4, 2, 3) \rightarrow (4, 5, 3) \rightarrow (4, 5, 6) \rightarrow (7, 5, 6) \rightarrow (7, 8, 6) \rightarrow (7, 8, 9) \rightarrow (7, 10, 9) \rightarrow (11, 10, 9) \rightarrow (11,10, 12) \rightarrow (13, 10, 12) \rightarrow (13, 14, 12) \rightarrow (13, 14,15) \rightarrow (16, 14, 15) \rightarrow (16, 17, 15) \rightarrow (16, 17, 18) \rightarrow (19, 17, 18) \rightarrow (19, 20, 18) \rightarrow (19, 20,21) \rightarrow (22, 20, 21) \rightarrow (22, 20, 23) \rightarrow (22, 24, 23) \rightarrow (25,24, 23) \rightarrow (25, 24, 26) \rightarrow (25, 27, 26) \rightarrow (28, 27, 26) \rightarrow (28,27, 29) \rightarrow T = (30, 27, 29)$. This sequence describes a 3-2-3 gait pattern, where successive 3-limb postures are interspersed by a 2-limb posture that allows motion of a limb between the two 3-limb postures. The path generated by the algorithm is minimal in terms of the number of foothold exchanges, where minimality is relative to the cube approximation of the feasible 3-limb postures (Figure 5).

6 Conclusion

We presented the PCG algorithm, for selecting the footholds of a 3-limb robot in planar tunnel environments with an arbitrary piecewise linear geometry. The algorithm assumes knowledge of the tunnel geometry and a lower bound on the amount of friction at the contacts. The algorithm approximates the collection of feasible 3-limb postures by p maximal cubes in each non-empty cell of contact c-space. Then it partitions the cubes and searches the sub-cube graph for the shortest 3-2-3 gait sequence from start to target. The algorithm's main strength is its emphasize on achieving contact independent foothold placement sequences. Each sub-cube parametrizes three contact independent wall segments, and each edge can be realized by limb lifting and re-placement between any two postures in the two sub-cubes connected by the edge. Thus a controller for the robot's limbs need only ensure footpad placement within the segments parametrized by the sub-cubes. The main weakness of the algorithm is the lack of a procedure for selecting the parameter p. This topic is under investigation.

Finally, it seems that the algorithm directly generalizes to k-limb mechanisms that move with a $k - (k - 1) - k$ gait pattern. Contact c-space in this case is k-dimensional, and one must first establish that the feasible k-limb postures in this space are a union of convex sets. If this is the case, the algorithm can be directly applied to such mechanisms. However, the computational complexity of the algorithm would become $O(np^{k+3} \log(np))$. A second more challenging topic is how to plan the footholds of a k-limb mechanism using a variable gait pattern.

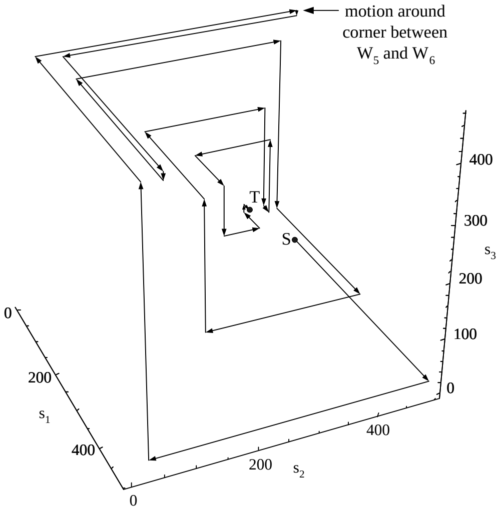

Figure 6: The shortest path from S to T along the edges of the sub-cube graph in contact c-space.

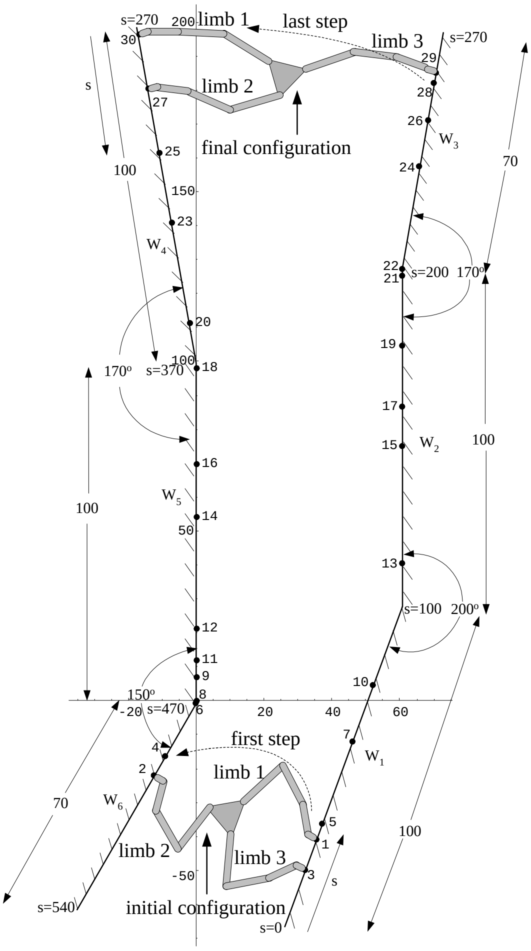

Figure 7: The tunnel environment used in the simulations, and the sequence of footholds generated by the PCG algorithm.

References

[1] J.-D. Boissonnat, O. Devillers, and S. Lazard. Motion planning of legged robots. *SIAM J. of Computing*, 30:218–246, 2000.

[2] S. Hirose and O. Kunieda. Generalized standard foot trajectory for a quadruped walking vehicle. *Int. J. of Robotics Research*, 10(1):2–13, 1991.

[3] J. Hong, G. Lafferriere, B. Mishra, and X. Tan. Fine manipulation with multifinger hands. *Icra*, 1568–1573, 1990.

[4] J. K. Lee and S. M. Song. Path planning and gait of walking machine in an obstacle-strewn environment. *J. Robotics Sys.*, 8:801–827, 1991.

[5] S. Leveroni and K. Salisbury. Reorienting objects with a robot hand using grasp gaits. *7th Int. Symp. on Robotics Research*, 2–15, 1995.

[6] A. Madhani and S. Dubowsky. Motion planning of mobile multi-limb robotic systems subject to force and friction constraints. *Icra*, 233–239, 1992.

[7] D. Marhefka and D. Orin. Gait planning for energy efficiency in walking machines. *Icra*, 474–480, 1997.

[8] V.-D. Nguyen. Constructing force closure grasps. *Int. J. of Robotics Research*, 7(3):3–16, 1988.

[9] F. Pfeiffer, T. Rossmann, N. Bolotnik, F. Chernousko, G. Kostin. Simulation and optimization of regular motions of a tube-crawling robot. *Multibody Sys. Dyn.*, 5:159–184, 2001.

[10] J. Ponce and B. Faverjon. On computing three-finger force closure grasps of polygonal objects. *IEEE Trans. on Robotics and Aut.*, 11(6):868–881, 1995.

[11] J. Savall, A. Avello, and L. Briones. Two compact robots for remote inspection of hazardous areas in nuclear power plants. *Icra*, 1993–1998, 1999.

[12] A. Shapiro and E. Rimon. Pcg: A foothold selection algorithm for spider robot locomotion in 2D tunnels. Tech. report, Dept. of ME, Technion, http://www.technion.ac.il/∼robots, July 2002.

[13] T. J. Stone, D. S. Cook, B. L. Luk. Robug III—an 8-legged teleoperated walking and climbing robot for disordered hazardous environments. *Mech. Incorp. Engineer*, 7(2):37–41, 1995.

[14] K. van der Doel and D. K. Pai. Performance measures for locomotion robots. *J. of Robotic Systems*, 14(2):135–147, 1997.

[15] Y. Wei and B. Goodwine. Stratified motion planning on non-smooth domains with application to robotic legged locomotion and manipulation. *Icra*, 3546–3551, 2002.