An Efficient Algorithm for One-Step Planar Compliant Motion Planning With Uncertainty

Amy J. Briggs*

TR 89-980
March 1989

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*This research has been supported in part by the NSF and MSI, Cornell University

An Efficient Algorithm for One-Step Planar Compliant Motion Planning With Uncertainty

Amy J. Briggs*

Department of Computer Science
Upson Hall
Cornell University
Ithaca, New York 14853

March 18, 1989

Abstract

Uncertainty in the execution of robot motion plans must be accounted for in the geometric computations from which plans are obtained, especially in the case where position sensing is inaccurate. We give an $O(n^2 \log n)$ algorithm to find a single commanded motion direction which will guarantee a successful motion in the plane from a specified start to a specified goal whenever such a one-step motion is possible. The plans account for uncertainty in the start position and in robot control, and anticipate that the robot may stick on or slide along obstacle surfaces with which it comes in contact. This bound improves on the best previous bound by a quadratic factor, and is achieved in part by a new analysis of the geometric complexity of the backprojection of the goal as a function of commanded motion direction.

*This research has been supported in part by the National Science Foundation and the Mathematical Sciences Institute, Cornell University.

1 Introduction

Motion plans for real robots must account for errors during execution. Consequently, given bounds on errors in sensing and control, we would like to plan motions that are guaranteed to succeed even in the worst case. *Uncertainty* as to control fundamentally changes the complexity of motion planning and the techniques employed. The introduction of uncertainty leads naturally and necessarily to allowing the robot to contact and comply with obstacle surfaces, because doing so greatly enriches the set of problems which can be solved.

We address the concrete and basic problem of finding a single commanded motion direction to maneuver a point robot from an uncertain start position in the plane to a specified goal where the robot is guaranteed to stop. Motions are strictly translational; thus the problem, for example, of modeling the geometric interactions of a peg and a hole can be reduced to navigating a point in configuration space [Loz83]. As in the classical motion planning problem, we assume the problem is posed in an environment of planar polygonal obstacles that is known and can be modeled exactly. The realization of a command, however, is subject to uncertainty since robots have imprecise sensing and imperfect control and therefore can only execute commands to within a given

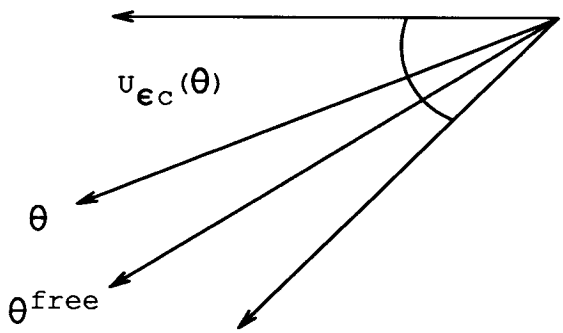

Figure 1: Uncertainty cone $U_{\epsilon c}(\theta)$.

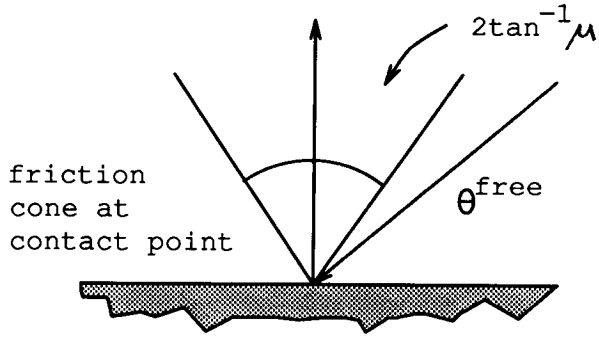

Figure 2: Sliding occurs at motion direction θ^{free}.

accuracy. Given a bound ϵc on the control error, we model this error as a cone in configuration space about a vector in the direction of the commanded motion θ and call it the *uncertainty cone* $U_{\epsilon c}(\theta)$ (see Figure 1). While executing a motion plan, the robot complies with the environment and may choose any direction consistent with the commanded direction and the control uncertainty. The direction chosen may vary over the execution of the plan. In what follows, n denotes the number of vertices in the environment and E denotes the number of edges in the visibility graph, where we assume $E = \Omega(n)$.

To deal with uncertainty in control, we allow *compliance* on obstacle surfaces, where a *compliant* motion is one during which the robot may slide or stick on obstacle surfaces. We model this effect by assuming generalized damper dynamics [Whi77,Don88] and Coulomb friction at point contacts, where the coefficient of friction μ is known and remains fixed for the environment. To determine if an obstacle surface is a sticking surface at motion direction θ, we check whether a vector at direction θ intersects the negative friction cone at the point of contact. If the vector lies outside the negative friction cone, then sliding will occur, otherwise sticking may occur (see Figure 2). In this model, we take a worst case approach; that is, we assume that if sticking is possible at a point, then the motion plan must prevent the point from being reached unless the point is in the goal.

Our method involves the construction of a concise representation for a structure called the *nondirectional backprojection* of the goal [Erd84, Don88]. By analyzing the changes to the backprojection as the motion direction varies, we give a complete characterization of the nondirectional backprojection and its complexity, and an efficient representation. To achieve this result, we develop an amortization strategy that is powerful enough to bound the number of changes to the boundary of the backprojection by $O(n^2)$. Hence, we can tighten the previous bound [Don88] on its combinatorial complexity from $O(n^3)$ to $O(n^2)$, and improve the algorithm for computing it from $O(n^4 \log n)$ to $O(n^2 \log n)$. As our examples show, the non-directional backprojection is a complex object which can undergo enormous global changes at a single event while remaining locally monotonic. We obtain our result by characterizing this local monotonicity.

We state the problem as follows:

Definition 1 *Given a planar polygonal environment $\mathcal{P}$ with constant size start region R and goal G, the* one-step planar compliant motion planning problem *is to find a commanded motion direction θ such that any trajectory from R consistent with the control uncertainty ϵc is guaranteed to reach G. The path should avoid obstacles or comply with the environment.*

1.1 Previous work

Canny and Reif [CR87] have shown that, in three dimensions, the problem of one-step motion planning is $\mathcal{NP}$-hard and the problem of multi-step motion planning is $\mathcal{NEXPTIME}$-hard. Previously, Natarajan had shown the multi-step

problem to be $\mathcal{PSPACE}$-hard [Nat86]. In a recent result, Canny gives doubly-exponential upper bounds for the general multi-step problem [Can89]. Our algorithm extends and improves the $O(n^4 \log n)$ algorithm by Donald [Don88] for finding translational motion direction θ in the plane.

1.2 Preliminaries

Given goal G, and commanded direction θ, the *backprojection* $B_\theta(G)$ is the set of all initial positions such that any trajectory consistent with the control uncertainty is guaranteed to reach the goal. Donald and Erdmann show that for constant size G, $B_\theta(G)$ can be computed in $O(n \log n)$ time using plane sweep techniques. Erdmann's algorithm is as follows [Don89,Erd86]:

1. For each non-goal vertex, determine whether the inverted control uncertainty cone $U_{\epsilon c}(\theta)$ intersects the friction cone at that vertex. If so, call this vertex a *sticking* vertex under commanded motion θ.

2. On each sticking vertex, erect two constraint rays parallel to the edges of the inverted control uncertainty cone.

3. Compute the arrangement of the environment with these $O(n)$ additional constraint rays.

4. Starting at a point in the goal, trace out the backprojection region (see Figure 3).

We call the maximal initial segment in free space of a constraint ray erected on a sticking vertex a *free space edge*. A *free space edge* e is specified by the sticking vertex p (called the *anchor vertex* of e) to which it is incident and an angle θ, where the anchor vertex remains fixed for e while θ varies. A free space edge is *vgraph critical* when it lies coincident with an edge of the visibility graph and therefore joins two obstacle vertices in free space. If free space edge r anchored at p and free space edge l anchored at $q \neq p$ intersect, we call the point of their intersection a *free space point*. Free space points are

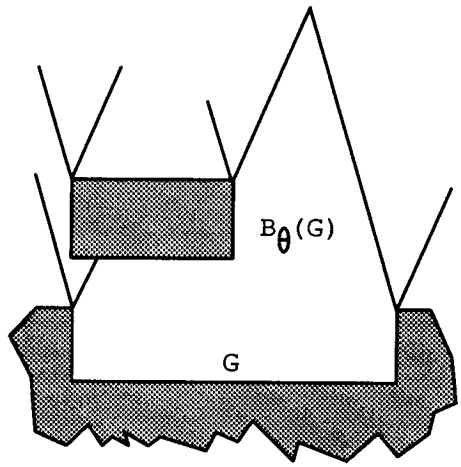

Figure 3: Backprojection construction.

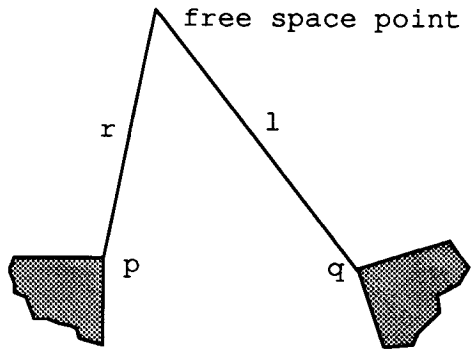

Figure 4: Free space point generated by intersection of two free space edges.

uniquely determined by their generating edges (see Figure 4).

Lemma 1 *For any direction θ and goal G of constant size, the backprojection $B_\theta(G)$ has size $O(n)$.*

Proof: The environment $\mathcal{P}$ has n obstacle edges and n vertices, which in turn contribute $O(n)$ constraint rays when cones are erected on all the sticking vertices. The backprojection is built from obstacle edges and free space edges, where a free space edge is a constraint ray anchored at an obstacle vertex and intersecting an obstacle edge or another constraint ray. Since at most two free space edges are anchored at each obstacle vertex, the backprojection has size at most $O(n)$. □

In order to find a commanded motion direction θ for which all trajectories from R are guaranteed to reach G, we evaluate the predicate $R \subset B_\theta(G)$ for selected values of θ. As we shall see, it suffices to consider those values of θ, called *critical values*, at which the topology of the backprojection changes. To this end, it is convenient to index the backprojection $B_\theta(G)$ with the critical motion direction θ at which it arises. This leads to the definition of the *non-directional backprojection* $B(G)$ as the set in $\Re^2 \times S^1$

$$B(G) = \bigcup_\theta (B_\theta(G) \times \{\theta\}).$$

Given that the topology of the backprojection does not change between critical events, Donald's approach [Don88] is to build $B(G)$ by computing a backprojection slice at each critical θ, and then test $B(G)$ for intersection with the start region. His algorithm builds a representation of size $O(n^3)$ for the non-directional backprojection in time $O(n^4 \log n)$. To obtain his result, Donald shows that we can bound the number of motion directions at which the topology of the backprojection changes, and thereby obtain a polynomial-sized representation for the non-directional backprojection. Since our proof allows this bound to be improved from $O(n^2)$ to $O(E)$, we give our own rendering.

Lemma 2 *There are $O(E)$ motion directions at which the topology of the backprojection changes.*

Proof: Note that the topology of the backprojection may change when any of the following events occurs:

Sliding critical event The determination of sliding vs. sticking on an obstacle edge changes due to a change in the motion direction θ.

Vgraph critical event A constraint edge becomes coincident with a visibility edge.

Vertex critical event A free space vertex of the backprojection coincides with an obstacle edge.

Since the topology of the backprojection can change only when an edge is inserted or deleted, these are the only events which can cause a topological change.

Sliding critical events contribute $O(n)$ critical values of θ since the determination of sliding vs. sticking on an edge can change at most 4 times. The visibility graph has $O(E)$ edges, and since all constraint edges are parallel to one or the other of the two edges of the control uncertainty cone, there are $O(E)$ vgraph critical values of θ.

Vertex critical events contribute $O(E)$ critical values since we can charge each vertex critical event to a unique visibility edge or sliding event as follows. Suppose the free space point determined by right edge $r(\theta_0)$ anchored at obstacle vertex p, and left edge $l(\theta_0)$ anchored at q, lies on obstacle edge e. Let x be the endpoint of e such that line (p, x) is at angle $\theta_1 \leq \theta_0$ ($r(\theta_0)$ has passed over x). Then one of the following must have occurred (see Figure 5):

1. Edge $r(\theta_1)$, $\theta_1 < \theta_0$, was coincident with the visibility edge between p and x.

2. x is not visible from p and edge $l(\theta_2)$, $\theta_1 < \theta_2 < \theta_0$, was coincident with a visibility edge between q and some vertex on an obstacle between p and x.

3. No such vgraph event as in 1. or 2. occurred, in which case at least one of p and q has become a sticking vertex at θ_3, $\theta_1 < \theta_3 < \theta_0$.

We assume that the environment is in general position, that is, at most one critical event will occur at any θ. Under this assumption, one of these events will occur first. In other words, we can choose θ_i, $1 \leq i \leq 3$, for which $\theta_0 - \theta_i$ is minimized. Thus we can charge the vertex event to a visibility edge or sliding event at θ_i. At most one other vertex event will be charged to the critical event at θ_i since, without any intervening sliding or vgraph events, a free space point travels in a piecewise circular arc and can coincide with an obstacle edge at most twice. We have shown above that the number of sliding and vgraph events is bounded by $O(E)$, so it follows that

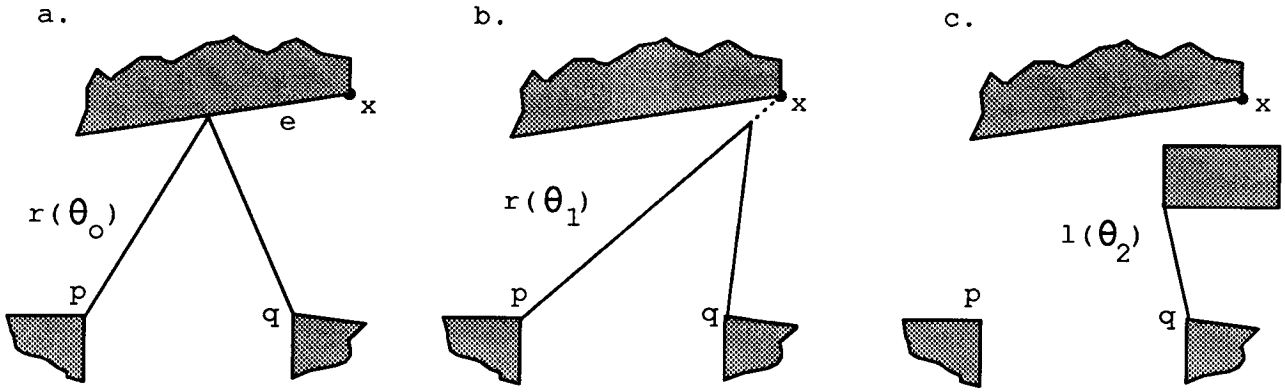

Figure 5: Vertex critical event is charged to a visibility edge.

the number of vertex critical events is bounded by $O(E)$. □

2 Main Result

As θ changes, the topology of the backprojection changes when edges are inserted or deleted at critical events. Using amortization techniques, we show that over the entire range of θ, the number of topological changes to the boundary of the backprojection is bounded by $O(n^2)$. Thus we can compute a representation for the non-directional backprojection incrementally instead of computing a slice for each critical θ. Namely, we fix $\theta_0 = 0$ and compute the ordered set

$$\{B_{\theta_0}(G), \Delta B_{\theta_1}(G), \ldots, \Delta B_{\theta_\omega}(G)\}$$

where $\theta_i \in \{\theta | \theta \text{ is critical}\}$, $\theta_0 < \theta_1 < \cdots < \theta_\omega$, and

$$\begin{aligned} \Delta B_{\theta_i}(G) \;=\; & (B_{\theta_{i-1}}(G) - B_{\theta_i}(G)\,, \\ & B_{\theta_i}(G) - B_{\theta_{i-1}}(G))\,, \end{aligned}$$

the net change to backprojection slice $B_{\theta_{i-1}}(G)$ to obtain backprojection slice $B_{\theta_i}(G)$ by adding and deleting edges. We show below that our representation for $B(G)$ has size $O(n^2)$ and is computed in time $O(n^2 \log n)$.

2.1 Computing the Non-Directional Backprojection

In this section we prove the following:

Theorem 1 *Given a goal G of constant size and an arrangement of input polygons $\mathcal{P}$ of size $O(n)$, a representation of size $O(n^2)$ for the non-directional backprojection $B(G)$ can be computed in time $O(n^2 \log n)$.*

Recall that under the assumption of general position, at most one critical event will occur at motion direction θ. Under this assumption, however, it is not true that a critical event causes only one topological change or only constant topological change to the backprojection. Indeed, we can construct examples where $O(n)$ changes occur at some θ, or where $n^{3/2}$ critical events occur, each causing $\sqrt{n}$ changes to the backprojection (see Figures 6 and 7). But with the following lemmas, we show that the total number of changes to the backprojection over the full range of θ is bounded by $O(n^2)$.

In what follows, we make use of an object called a *conforming sweep line*. The idea is similar to that of a topological sweep line [Ede87]. A conforming sweep line is specified by an anchor point q and an angle θ, and conforms to the environment in a way to be made precise shortly. A conforming sweep line divides the plane into two regions, *ahead* and *behind*. We say that configuration space point x is *ahead* of a conforming sweep line at motion direction θ_1 if the sweep line will cross x at $\theta \geq \theta_1$. Similarly, x is *behind* a conforming sweep line at motion direction θ_1 if the sweep line crossed x at some $\theta \leq \theta_1$. As θ increases and critical events occur, we imagine that a conforming sweep line changes continu-

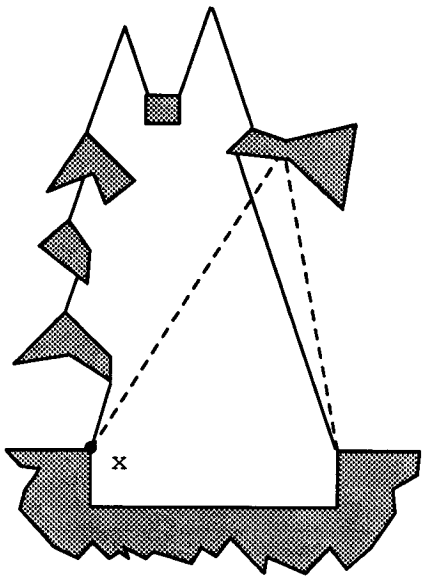

Figure 6: Large change in backprojection caused by vgraph event at vertex x.

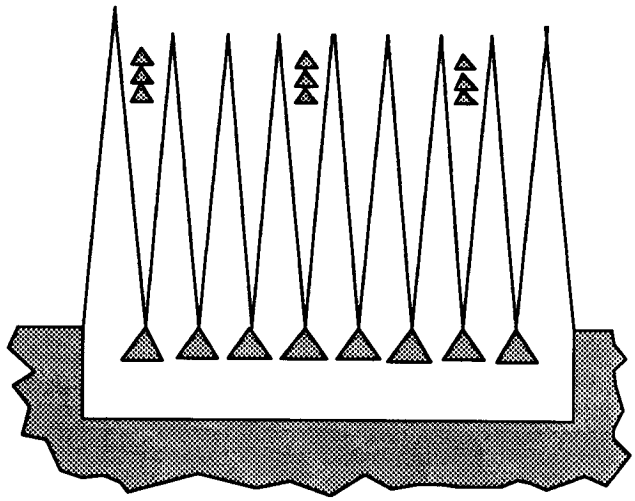

Figure 7: Each of the $O(n)$ spikes of the backprojection is vgraph critical with each of the $O(\sqrt{n})$ clusters. Each cluster is of size $O(\sqrt{n})$, so each of $O(n^{3/2})$ vgraph critical events causes $O(\sqrt{n})$ changes to the topology of the backprojection.

ously, sweeping out area between critical events. If configuration space point $x \in B_\theta(G) - B_{\theta-\delta}(G)$ as $\delta \to 0$ then we say x *enters* the backprojection at θ. This will occur when some conforming sweep line sweeps over x in its continuous motion (see Figure 8). Similarly, x *leaves* the backprojection at θ if $x \in B_{\theta-\delta}(G) - B_\theta(G)$.

We employ the convention that as θ increases monotonically over the range $[0, 2\pi)$, the corresponding control uncertainty cone $U_{\epsilon c}(\theta)$ rotates in a counterclockwise direction, which has the effect of locally "rotating" the backprojection in a counterclockwise direction. A *left velocity cone edge* $l(\theta)$ is a free space edge lying parallel to the left edge of the inverted control uncertainty cone $-U_{\epsilon c}(\theta)$ for a particular motion direction θ. A *right velocity cone edge* $r(\theta)$ is defined analogously.

Definition 2 *Let free space edge $e(\theta)$, anchored at obstacle vertex q, lie at angle $\theta_0 = \theta \pm \epsilon c$. A* conforming sweep line *on $e(\theta)$, denoted $csl_\theta(e)$, is defined for a maximal interval of $\theta \geq \theta_0$ in which no sliding event occurs. It is composed of two semi-infinite segments. One is fixed at initial angle θ_0 and extends from q at angle $\pi + \theta_0$, that is, in the direction opposite to $e(\theta)$. The other segment has the following recursive defi-*

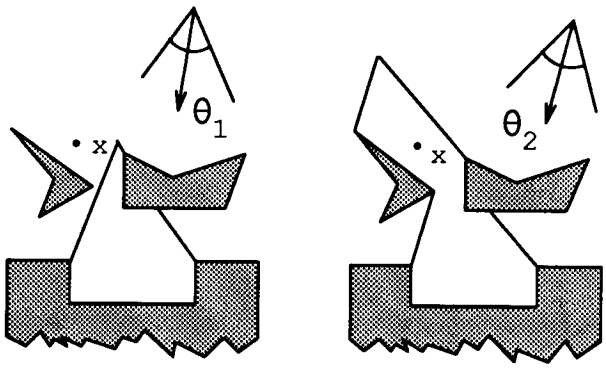

Figure 8: Configuration space point x enters the backprojection at $\theta_2 > \theta_1$.

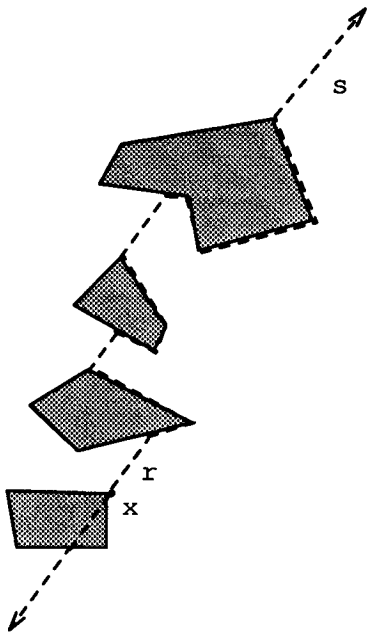

Figure 9: Conforming sweep line s on r anchored at x.

nition. Extend e at angle θ_0 until it intersects an obstacle. Follow the boundary of the obstacle counterclockwise until the first vertex q' that is a sticking vertex for direction $\theta + \pi$. Treat this vertex as q and continue (see Figure 9).

Sweep lines are monotonic, that is, once a point is behind a conforming sweep line, it stays behind that conforming sweep line. Since we exploit this monotonicity in what follows, we make the notion precise:

Proposition 1 *Let $l(\theta_1)$ be a left velocity cone edge anchored at q, where $l(\theta_1)$ lies on the boundary of the backprojection. Let $r(\theta_1)$ be a right velocity cone edge anchored at p where p is ahead of $csl_{\theta_1}(l)$. $csl_\theta(r) \cap B_\theta(G)$ lies ahead of $csl_\theta(l)$ for $\theta \geq \theta_1$ for a maximal interval in which no sliding event occurs.*

Lemma 3 *Let $l(\theta_1)$ be a left velocity cone edge anchored at q, and consider the conforming sweep line $csl_{\theta_1}(l)$. No point behind $csl_{\theta_1}(l)$ will enter the backprojection due to a vgraph critical event for a vertex ahead of $csl_{\theta_2}(l)$ for $\theta_2 > \theta_1$ unless a sliding event occurs between θ_1 and θ_2.*

Proof: Let x be a point in configuration space such that x enters $B_\theta(G)$ for $\theta > \theta_1$. $x \in B_{\theta+\delta}(G) - B_{\theta-\delta}(G)$, which is ahead of $csl_\theta(l)$ by the above proposition. Thus x lies ahead of $csl_\theta(l)$. □

Lemma 4 *Suppose free space edge $r(\theta_1)$ is vgraph critical and causes free space point x to enter the interior of the backprojection. As θ increases, if x leaves the interior of the backprojection at $\theta_2 > \theta_1$, then edge $r(\theta_3)$ at $\theta_3 > \theta_2$ cannot cause x to re-enter the interior of the backprojection by being vgraph critical with another vertex as long as no sliding event occurs between θ_2 and θ_3.*

Proof: Observe that only right ray events cause new vertices to become part of the backprojection, and only left ray events cause vertices to leave the backprojection (see Figure 10). Let $r(\theta)$ be a right edge anchored at obstacle vertex p such that when $r(\theta_1)$ is vgraph critical with obstacle vertex v, free space point x enters the backprojection. Suppose a vgraph event at θ_2 causes x to leave the interior of the backprojection. x lies behind the conforming sweep line on $r(\theta_1)$ and this line is behind $csl_{\theta_2}(r)$ at $\theta_2 > \theta_1$, so x lies behind the conforming sweep line on $r(\theta_2)$. But since $x \notin B_{\theta_2}(G)$, x must lie behind $csl_{\theta_2}(l)$ for some left edge $l(\theta_2)$. We know p lies ahead of $csl_{\theta_2}(l)$, so by the lemma above, x cannot re-enter the backprojection due to a vgraph critical event involving r anchored at p. □

We are now ready to prove the following:

Lemma 5 *There are $O(n^2)$ changes to the topology of the backprojection over all values of θ.*

Proof: With the above lemmas we have shown that a free space edge can cause a free space point to enter the interior of the backprojection at most once. Since an obstacle edge segment can enter the boundary of the backprojection only when some adjacent free space enters the backprojection, it follows that a free space edge can cause a nontrivial obstacle edge segment to enter the boundary at most once. Note

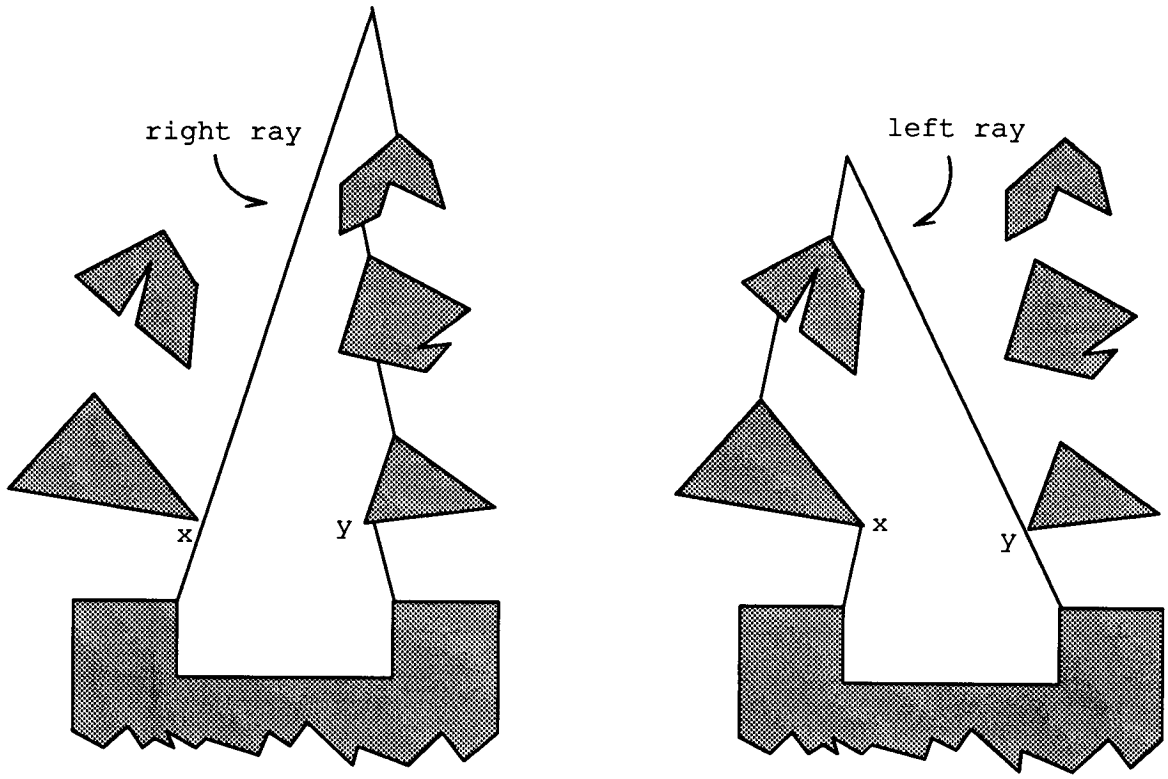

Figure 10: Changes to backprojection due to left and right ray critical events at vertices x and y.

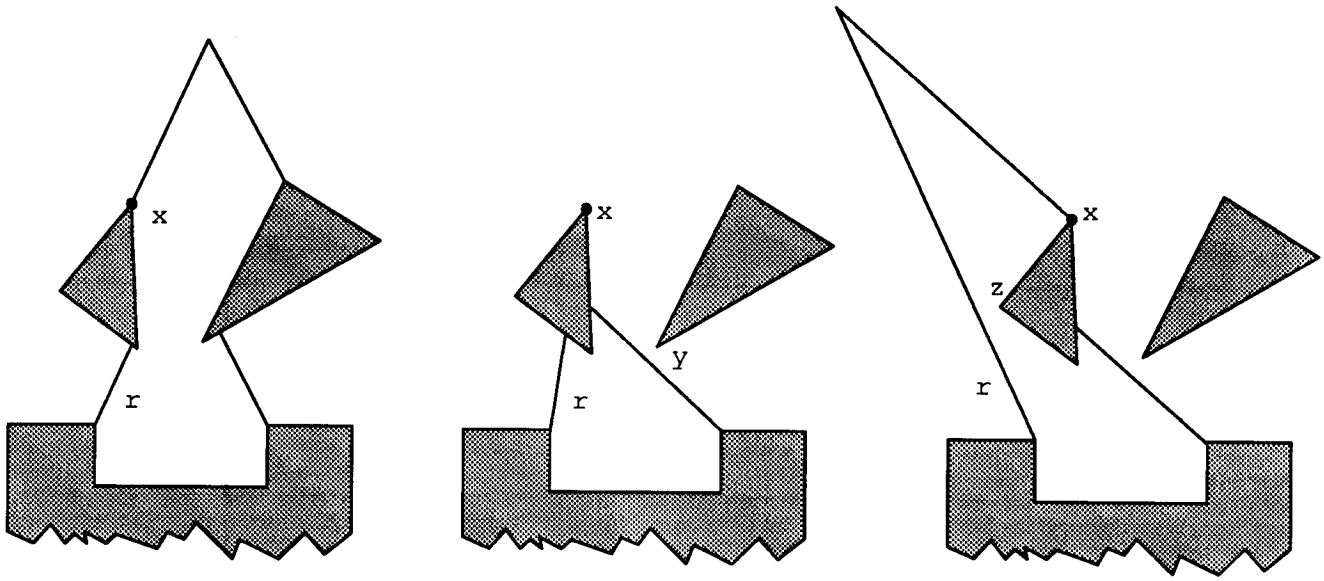

Figure 11: Free space edge r causes obstacle vertex x to enter the boundary of the backprojection twice over all values of θ.

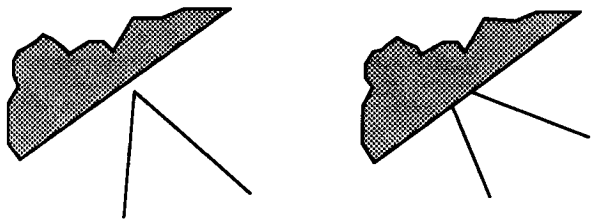

Figure 12: Changes to boundary of backprojection due to vertex critical events.

that we assume no intervening sliding events between the vgraph events which would bring a non-vertex point back into the backprojection. This assumption is valid since if a non-vertex configuration space point x enters the backprojection twice due to edge e being vgraph critical and a sliding event occurs between the two instances, then we can charge the re-entry of x to the sliding event, since there are only $O(n)$ sliding events. With the assumption of general position, we can make the observation that an obstacle vertex becomes part of the boundary of the backprojection only if an adjacent obstacle edge segment also becomes part of the boundary. An obstacle vertex has two adjacent edges, so a free space edge can cause an obstacle vertex to enter the boundary of the backprojection at most twice over all θ (see Figure 11). There are $O(n)$ free space edges, and n obstacle vertices, so this gives $O(n^2)$ changes to the topology of the backprojection over all vgraph critical events.

Sliding events contribute only $O(n)$ critical values of θ, so the total number of changes to the topology of the backprojection due to sliding events is $O(n^2)$. A vertex critical event occurs when a free space point coincides with an obstacle edge. Since the environment is assumed to be in general position, no new constraint edges arise and therefore the only change to the boundary of the backprojection is that an obstacle edge segment is inserted or deleted (see Figure 12). Thus vertex critical events cause only local changes to the boundary of the backprojection, so they contribute only $O(n^2)$ topological changes.

Thus the total number of changes to the topology of the backprojection is bounded by $O(n^2)$. □

3 The Algorithm

The original arrangement of obstacles has size n and is taken as input. The visibility edges and *pseudo-visibility edges* between all pairs of vertices are then computed in time $O(n^2)$ [AAG+86] where n is the number of vertices in the environment. The visibility graph and arrangement remain fixed throughout the computation, which then begins by fixing θ at an initial value, say $\theta_0 = 0$, and computing a backprojection $B_{\theta_0}(G)$. To keep the backprojection updated as θ changes, a priority queue for each type of critical value is maintained. These queues are initialized as follows:

- Sliding critical values: For each obstacle edge e, find each θ at which the determination of sliding vs. sticking on e will change. Insert the pairs (θ, e) in the sliding critical queue, ordered by θ.
- Vgraph critical values: For each constraint edge e, find each θ at which e will become coincident with a visibility edge. Insert the pairs (θ, e) in the vgraph critical queue, ordered by θ. The visibility graph, which has size $O(E)$, can be found and the critical values sorted in time $O(n^2 \log n)$.
- Vertex critical values: For purposes of this discussion, we say that free space backprojection edges e_1 and e_2 are *consecutive* if e_1 is coincident with a right velocity cone edge, and e_2 is the next free space edge along the backprojection, where e_2 is coincident with a left velocity cone edge. Then for each pair of consecutive free space edges on the backprojection, find the first θ at which their intersection point p will lie on an obstacle edge. Insert the pair (θ, p) in the vertex critical queue, ordered by θ.

Since each queue contains $O(n^2)$ elements, each requires time $O(n^2 \log n)$ to initialize and $O(\log n)$ to update.

Note that the sliding and vgraph critical values can be computed ahead of time, while the vertex critical values cannot since we do not know

where the free space points will be. To solve this problem, we introduce a data structure to help keep track of potential vertex events. Recall that a free space edge is the maximal initial segment of a semi-infinite constraint ray. As the computation proceeds, for each constraint edge e, we keep an updated priority queue of obstacles which the ray extension of the edge intersects. If ray e, anchored at p, lies coincident with vertex v of obstacle j, then the edges of j incident to v are added to or deleted from the priority queue for e depending on whether e will intersect them as θ increases. Since there are $O(n)$ constraint rays and $O(n)$ obstacle edges, the overall time to maintain these queues is $O(n^2 \log n)$. Then at each vgraph or sliding event, we find the new free space points and calculate their impending intersections with obstacles, based on the priority queues for their generating edges. This can be done by computing the arc in which the free space point travels and intersecting it with the common edges in the two queues. These potential vertex events are added to the vertex critical event queue, and the computation continues. As we showed earlier, there is only constant amortized change to the boundary of the backprojection for vgraph events, so there is only a constant amortized number of new free space points due to a single vgraph event.

Suppose we have some slice of the non-directional backprojection computed and wish to compute the next backprojection slice. The next critical value of θ can be found by comparing the first elements of the three priority queues, and removing the minimum valid event. Note that an event is valid if it arises from a vertex or edge that is part of the backprojection; not all queued events will be valid. If the next event arises from a sliding criticality, then a new backprojection slice can be computed using the plane sweep algorithm, since there are only $O(n)$ sliding critical values. In the case of a vertex critical value, the only change to the backprojection will be that a free space vertex either enters or leaves the boundary of an obstacle. The backprojection boundary can then be updated locally to reflect the change. On the other hand, a vgraph criticality can cause $O(n)$ changes to the backprojection (see Figure 6). Since there are potentially $O(n^2)$ vgraph critical values, we would like to update the backprojection incrementally when one arises. This can be accomplished by deleting the critical edge e from the backprojection and then tracing out the new backprojection region starting at one endpoint of e until some vertex on $B_\theta(G)$ is reached. The following lemma shows that this can be accomplished in time linear in the size of the change.

Lemma 6 *Given polygonal environment $\mathcal{P}$, goal G, commanded motion direction θ, control uncertainty ϵc, and backprojection vertex u, vertex v adjacent to u on the backprojection $B_\theta(G)$ can be found in $O(1)$ time, making use of a data structure which costs $O(n^2 \log n)$ to maintain over the entire computation.*

Proof: For each sticking vertex v, we keep a priority queue of free space rays which intersect the right ray of v, ordered by the distance from v to the ray intersection. Each time a new free space edge is created or deleted at a sliding critical event, we update the priority queues. Since there are $O(n)$ rays, and each is inserted into $O(n)$ queues at a cost of $O(\log n)$ per insert, the overall time to maintain the data structure is $O(n^2 \log n)$. □

3.1 Algorithm One-Step

Once the non-directional backprojection $B(G)$ has been computed, we can check for containment of the start region R. $R \subset \mathcal{P}$ has a constant number of vertices, some or all of which may be in free space, so we can generate a *pseudo-critical* event when an edge of R intersects $B_\theta(G)$. If a *pseudo-critical* event occurs at θ_i, then we test the backprojection at $B_{\theta_i}(G)$ for containment of R. To do this, we maintain an updated backprojection slice

$$\begin{aligned} S_i = S_{i-1} \quad &\cup \quad \{B_{\theta_i}(G) - B_{\theta_{i-1}}(G)\} \\ &- \quad \{B_{\theta_{i-1}}(G) - B_{\theta_i}(G)\} \end{aligned}$$

where $\theta_i \in \{\theta | \theta \text{ is critical}\}$ and test each S_j, $j \in \{j | \theta_j \text{ is pseudo-critical}\}$. If θ is found such that

$R \subset B_\theta(G)$, then motion direction θ from R is guaranteed to reach the goal. Otherwise, no one-step motion is guaranteed to reach G.

Recall that our representation for $B(G)$ is of size $O(n^2)$ and consists of $O(n^2)$ partial backprojection slices. Since we can test for containment of R in $O(\log n)$ time per slice [Don88], we have established the following:

Theorem 2 *The one-step planar compliant motion planning problem with uncertainty can be solved in time $O(n^2 \log n)$.*

4 Conclusion

In this work we give an $O(n^2 \log n)$ algorithm for the basic problem of manipulating a point from a specified start region to a specified goal region amidst planar polygonal obstacles where control is subject to uncertainty. This result represents a quadratic improvement over the best previous bounds and is achieved by a new analysis of the geometric complexity of the non-directional backprojection and an efficient algorithm for computing its representation.

4.1 Acknowledgements

I am immensely grateful to Donald Johnson and to my advisor Bruce Donald for their enthusiasm and encouragement and help in so many ways. I also thank John Canny, Joe Mitchell, and Randy Brost for helpful discussions and comments.

References

[AAG+86] T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai. Visibility of disjoint polygons. *Algorithmica*, 1:49–63, 1986.

[Can89] J. Canny. On computability of fine motion plans. In *IEEE International Conference on Robotics and Automation, Phoenix, Az.*, May 1989.

[CR87] J. Canny and J. Reif. New lower bound techniques for robot motion planning problems. In *Proc. IEEE Symp. on Foundations of Computer Science*, 1987.

[Don88] B. R. Donald. The complexity of planar compliant motion planning under uncertainty. In *Proc. ACM Symp. on Computational Geometry, Urbana*, June 1988.

[Don89] B. R. Donald. *Error detection and recovery in Robotics*, volume 336 of *Lecture Notes in Computer Science.* Springer-Verlag, Berlin, 1989.

[Ede87] H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* Springer-Verlag, Berlin, 1987.

[Erd84] M. Erdmann. On motion planning with uncertainty. Technical Report MIT–AI–TR 810, MIT Artificial Intelligence Laboratory, 1984.

[Erd86] M. Erdmann. Using backprojections for fine motion planning with uncertainty. *IJRR*, 5(1), 1986.

[LMT84] T. Lozano-Pérez, M. T. Mason, and R. H. Taylor. Automatic synthesis of fine-motion strategies for robots. *Int. J. of Robotics Research*, 3(1), 1984.

[Loz83] T. Lozano-Pérez. Spatial planning: A configuration space approach. *IEEE Trans. on Computers (C–32)*, pages 108–120, 1983.

[Mas82] M. T. Mason. Manipulator grasping and pushing operations. Technical Report MIT AI–TR–690, MIT AI Lab, 1982.

[Mas84] M. T. Mason. Automatic planning of fine motions: Correctness and completeness. In *IEEE International Conference on Robotics, Atlanta Ga.*, 1984.

[Nat86] B. K. Natarajan. *On Moving and Orienting Objects.* PhD thesis, De-

partment of Computer Science, Cornell University, Ithaca, N.Y., 1986.

[NP82] J. Neivergelt and F. P. Preparata. Plane-sweep algorithms for intersecting geometric figures. *CACM*, 25(10), 1982.

[Whi77] D. Whitney. Force feedback control of manipulator fine motions. *Journal of Dynamic Systems, Measurement, and Control June*, pages 91–97, 1977.