On Finding Narrow Passages with Probabilistic Roadmap Planners

David Hsu, *Stanford University, Stanford, CA, USA*
Lydia E. Kavraki, *Rice University, Houston, TX, USA*
Jean-Claude Latombe, *Stanford University, Stanford, CA, USA*
Rajeev Motwani, *Stanford University, Stanford, CA, USA*
Stephen Sorkin, *Stanford University, Stanford, CA, USA*

A probabilistic roadmap is a network of simple paths connecting collision-free configurations obtained by sampling a robot's configuration space at random. Several probabilistic roadmap planners have solved unusually difficult path-planning problems, but their efficiency remains disappointing when the free space contains narrow passages. This paper provides foundations for understanding the effect of passages on the connectedness of probabilistic roadmaps. It also proposes a new random sampling scheme for finding such passages. An initial roadmap is built in a "dilated" free space allowing some penetration distance of the robot into the obstacles. This roadmap is then modified by resampling around the links that do not lie in the true free space. Experiments show that this strategy allows relatively small roadmaps to reliably capture the free space connectivity.

1 Introduction

Probabilistic roadmap planners (PRMs) construct a network of simple paths (usually straight paths in configuration space) connecting collision-free configurations picked at random [2, 4, 5, 10, 11, 13, 16, 24, 27]. They have been successful in solving difficult path-planning problems, but their efficiency remains disappointing when the free space contains narrow passages. In [4, 10, 14, 15], we have formally investigated the performance of PRMs. The property of ϵ-goodness [15] allows us to determine how well a probabilistic roadmap "covers" the free space, while the properties of expansiveness [10] and path clearance [14] allow us to analyze the connectedness of a roadmap.

Exploiting these results we now motivate a new sampling strategy to find paths through narrow passages more efficiently than with previous strategies. An initial roadmap R' is computed in a "dilated" free space F' allowing some penetration distance of the robot into the obstacles. Next, the milestones and links of R' that do not lie in the free space F are "pushed" into F by local resampling operations, in order to construct a final roadmap R. The underlying intuition is that, by widening narrow passages, dilatation improves ϵ-goodness, expansiveness, and path clearance, so that the connectivity of F' is easier to capture than that of F. Preliminary experiments have produced very encouraging results.

Path planners using techniques to expand or shrink obstacles and/or robots have previously been proposed [3, 9], but with different planning approaches.

Section 2 describes a "basic" PRM. Section 3 surveys other sampling strategies proposed in the literature. Section 4 establishes the relation between the results in [4, 10, 14, 15] and the complexity of the basic PRM in the presence of narrow passages. Section 5 describes and discusses the new sampling strategy.

2 Basic PRM

Let $\mathcal{C}$ denote the configuration space of a robot and F the open subset of collision-free configurations, *i.e.*, the *free space*. For simplification, let $\mathcal{C}$ be the Euclidean hyper-cube $[0,1]^n$. We say that two free configurations *see* each other if they can be connected by a straight-line path in F.

The basic PRM is a simplified version of the planner in [16]. First, the procedure `roadmap` precomputes a roadmap R; then `query` uses R to process path-planning queries. Each query is defined by two free configurations, q_b and q_e, called the *query configurations*. A correct answer is a path connecting them in F, if one such path exists, and the indication that no such path exists, otherwise.

procedure `roadmap`:

1. Pick s configurations uniformly at random in F. Call them *milestones* and let M be the set of milestones.
2. Construct the graph $R = (M, L)$ in which L consists of every pair of milestones that see each other. Call R the *roadmap*.

Figure 1: *Roadmap-construction algorithm*

procedure `query`:

1. **for** $i = \{b, e\}$ do:
   (a) **if there exists** a milestone m that sees q_i **then** $m_i \leftarrow m$.
   (b) **else**
      i. **repeat** t **times:**
         Pick a configuration q in F uniformly at random in a neighborhood of q_i
         until q sees both q_i and a milestone m.
      ii. **if all** t trials failed **then return** FAILURE, **else** $m_i \leftarrow m$.
2. **if** m_b and m_e are in the same component of the roadmap, **then return** a path connecting them, **else return** NO-PATH.

Figure 2: *Query processing algorithm*

The procedure `roadmap` (Figure 1) chooses the milestones at Step 1 and creates the links between milestones at Step 2. Let `dist`(q) be a procedure that computes the Euclidean distance between the robot placed at q and the obstacles. Step 1 generates each milestone by picking successive configurations q in $\mathcal{C}$, until one satisfies `dist`(q) > 0. Every q is obtained by choosing each of its coordinates uniformly at random in $[0, 1]$. Step 2 checks the straight path between every two milestones for collision, by recursively decomposing it into two half segments and invoking `dist` at each segment endpoint [4].

The procedure `query` (Figure 2) tries to connect each query configuration to a milestone of the roadmap, either directly (Step 1(a)), or through an intermediate configuration chosen in a neighborhood of the query configuration (Step 1(b)), using `dist` to check connections for collision. Each free configuration q at Step 1(b)i is obtained by picking successive configurations at random in a hyper-cube centered at q_i until one is collision-free. The procedure may output FAILURE at Step 1(b)ii (if it cannot connect a query configuration to some milestone of the roadmap) and NO-PATH at Step 2 (if it connects the query configurations to two distinct components of the roadmap). The NO-PATH answer may be incorrect if two components of the roadmap lie in the same component of F. Clearly, we would like the planner to rarely return FAILURE or an incorrect NO-PATH answer.

The cost of computing an explicit representation of F in a high-dimensional space is prohibitive. Instead, a PRM uses the implicit representation of F provided by `dist`, which only computes distances in $\mathbb{R}^2$ or $\mathbb{R}^3$ and admits several reasonably efficient implementations (*e.g.*, [7, 8, 12, 18, 19, 20, 22, 25]). In this paper we often use simple illustrative examples of 2-D free spaces, which could easily be handled by other planning techniques. Keep in mind that in practical problems, it is often not realistic to explicitly represent F.

3 Other Sampling Strategies

The planner in [5] uses a partially random sampling strategy. It deterministically follows the steepest descent of a heuristic potential field defined over $\mathcal{C}$ until it reaches a minimum of the potential. If this minimum is the goal, it stops; otherwise it tries to escape its basin of attraction by performing a series of random walks. Therefore, the potential field strongly biases the planner's sampling strategy. In some cases, it successfully guides the planner through narrow passages. But it may also have exactly the opposite effect, when a local basin of attraction contains a narrow passage.

The planner in [13, 16] includes a resampling step to improve the connectedness of the roadmap. An initial roadmap is generated using an algorithm similar to `roadmap`. Next, additional milestones are added in the neighborhoods of milestones that see no or few other milestones. Experiments have shown that this resampling scheme is very effective. But it still does not provide an efficient means of finding connections through narrow passages. Note that the rationale underlying this two-stage sampling strategy is similar to the one in this paper. In both cases, the first stage attempts to capture the connectivity of a free space using no a priori knowledge, while the second stage adds milestones in subsets of F chosen on the basis of the information revealed by the first stage.

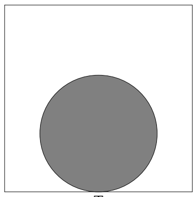
We will see that the sampling strategy proposed in this paper creates a greater density of milestones near the boundary of the free space. Other roadmap planners try to attain this goal using different strategies. In [2], when a configuration q is generated outside F, a number of rays are shot from q along random directions uniformly distributed in $\mathcal{C}$. For each ray, a binary search is used to identify a point near the boundary of F. In [23] a single ray is shot from q along a random direction; the procedure then simulates a walk of the robot along this direction, until it reaches free space. In [11] new milestones are created near F's boundary to connect roadmap components that could not be connected by straight paths. A ray is shot from a milestone in one component along a direction picked at random. Using a technique similar to [23], a milestone is created where this ray encounters the free space boundary, and the ray is reflected in a random direction at this point to find another boundary point. All three references listed above observe that adding milestones near the free space boundary improves the planners' performance.

Various sampling and connection strategies are described and experimentally evaluated in [1]. In particular, a multi-stage strategy is proposed which allows milestones to be connected by multiple types of paths.

4 Complexity of the basic PRM

To avoid the FAILURE outcome, a PRM must pick milestones that collectively see a large portion of F, so that any query configuration can easily be connected to one of them. To avoid incorrect NO-PATH replies, it must connect the milestones so that there is a one-to-one correspondence between the components of F and those of the roadmap.

In this section we analyze the complexity of the basic PRM by estimating the number s of milestones (the *size* of the roadmap) that must be generated in order to answer queries correctly with high probability. We relate this number to the extent to which F satisfies desirable geometric properties: ϵ-goodness [4, 15], expansiveness [10], and path clearance [13, 14]. Though not perfect, s is a reasonable measure of the work done by the planner.



T

Figure 3: *A free space that is not ϵ-good*

For any subset $S \subseteq \mathcal{C}$, $\mu(S)$ denotes its volume. For any $q \in F$, $\mathcal{V}(q)$ denotes the set of all free configurations seen by q; we call it the *visibility set* of q.

4.1 Roadmap Coverage

Definition 1. *Let ϵ be a constant in $(0,1]$. A free configuration q is* ϵ-good *if $\mu(\mathcal{V}(q)) \geq \epsilon\mu(F)$. The free space F is* ϵ-good *if every $q \in F$ is ϵ-good.*

There exist spaces that are not ϵ-good. This is the case of the 2-D free space shown in Figure 3. Every point q in this space is ϵ-good for some $\epsilon \in (0,1]$. However, if q tends toward the tangency point T, $\epsilon \rightarrow 0$.

Definition 2. *A set of milestones provides* adequate coverage *of an ϵ-good free space F if the volume of the subset of F not visible from any of these milestones is at most $(\epsilon/2)\mu(F)$.*

A greater ϵ will make it easier for `query` to connect query configurations to the roadmap. Hence, as ϵ increases, the coverage requirement grows weaker, *i.e.*, the portion of F that has to be visible by at least one milestone gets smaller.

Theorem 1. *Assume that F is ϵ-good. Let ϕ be a constant in $(0,1]$ and K be a positive real large enough that for any $x \in (0,1]$, $(1-x)^{(K/x)\log(2/x\phi)} \leq x\phi/2$. If s is chosen such that:*

$$s \geq \frac{K}{\epsilon}(\log\frac{1}{\epsilon} + \log\frac{2}{\phi}),$$

then `roadmap` *generates a set of milestones that adequately covers F, with probability at least $1-\phi$.*

See [4, 15] for a proof of this theorem.

Theorem 1 does not allow us to compute s since we do not know the value of ϵ, except for simple spaces. Nevertheless, it tells us that although adequate coverage of F is not guaranteed, the probability that inadequate coverages happens decreases exponentially

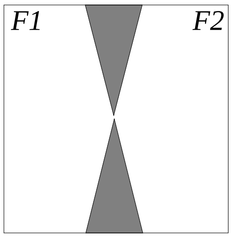


Figure 4: *A narrow passage in an ϵ-good space*

with the number of milestones. Moreover, the number of milestones needed increases moderately when ϵ decreases.

It now remains to establish that adequate coverage allows `query` to connect any query configuration to the roadmap, with high probability.

Theorem 2. *Let the maximum number of iterations t at Step 1(b)i of* `query` *be set to $\log(2/\psi)$, where ψ is a constant in $(0,1]$. If the milestones adequately cover F, then the probability that* `query` *outputs* FAILURE *is at most ψ.*

In other words, the probability that `query` returns FAILURE decreases exponentially with the number t of iterations at Step 1(b)i. See [15] for a proof of this theorem. This proof assumes that Step 1(b)i of `query` samples the visibility set $\mathcal{V}(q_i)$ of q_i to find a configuration q that sees both q_i and a milestone m. Since $\mathcal{V}(q_i)$ is unknown, any implementation of `query` is an approximation of the algorithm to which Theorem 2 strictly applies.

However, ϵ-goodness is too weak a requirement to guarantee that `roadmap` will construct a roadmap whose connectivity correctly represents that of the free space. For example, the free space of Figure 4 is ϵ-good for $\epsilon \approx 0.5$. But a roadmap constructed by `roadmap` will most likely consist of two connected components. This type of example leads us to introduce the notion of an expansive free space [10].

4.2 Roadmap Connectedness

Definition 3. *Let F be an ϵ-good free space. A roadmap R is an* adequate representation *of F if its milestones provide adequate coverage of F and no two components of R lie in the same component of F.*

Let R be an adequate representation of F. Since F is ϵ-good, no component of F has volume less than $\epsilon\mu(F)$. Therefore, at least one milestone of R lies in every component of F. Since no two components of R lie in the same component of F, there is a one-to-one correspondence between the components of R and those of F.

The notion of an expansive free space is directly related to the difficulty that `roadmap` has to connect milestones through narrow passages. The reason why it would require considerable time for `roadmap` to build a connected roadmap in the free space of Figure 4 is that a very small subset of points in F_1 see a large fraction of F_2; therefore, the probability that the planner picks a milestone in F_1 that sees a milestone in F_2 is small. By narrowing the passage between F_1 and F_2, one can make this probability arbitrarily small. We refer to the subset of points in a subset $S \subset F$ that can see a large portion of $F \backslash S$ as the lookout of S. If it is large enough, it is easy to connect any point in S to points outside S by picking points at random in S and $F \backslash S$.

Definition 4. *Let β be a constant in $(0,1]$ and S be any subset of any component E of the free space F. The β-*lookout *of S is the set:*

$$\beta\text{-LOOKOUT}(S) = \{q \in S \mid \mu(\mathcal{V}(q) \backslash S) \geq \beta\mu(E \backslash S)\}.$$

Definition 5. *Let ϵ, α, and β be constants in $(0,1]$. The free space F is (ϵ,α,β)-*expansive *if it is ϵ-good and, for every connected subset S, we have:*

$$\mu(\beta\text{-LOOKOUT}(S)) \geq \alpha\mu(S).$$

We abbreviate "(ϵ,α,β)-expansive" by "expansive".

Theorem 3. *Assume that F is (ϵ,α,β)-expansive. Let ξ be a constant in $(0,1]$. If s is chosen such that:*

$$s \geq \frac{16}{\epsilon\alpha}\log\frac{8}{\epsilon\alpha\xi} + \frac{6}{\beta} + 4,$$

then with probability at least $1-\xi$, `roadmap` *generates a roadmap such that no two of its components lie in the same component of F.*

Given a set M of milestones, let a *linking sequence* of a milestone m_0 be a sequence $m_1, m_2, \ldots$, such that for all $i > 1$, $m_i \in \text{LOOKOUT}(V_{i-1})$, with $V_0 = \mathcal{V}(m_0)$ and $V_i = V_{i-1} \cup \mathcal{V}(m_i)$. Let the visibility set of a linking sequence be the union of the visibility sets $\mathcal{V}(m_i)$ of all the milestones m_i in the sequence. The proof of Theorem 3, given in [10], first establishes that in a properly sized roadmap, for every pair of milestones (m, m') in

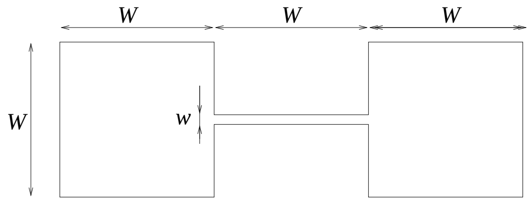


Figure 5: *An expansive free space where $\epsilon, \alpha, \beta \sim w/W$*

the same component of F, with high probability m admits a linking sequence whose visibility set intersects that of a linking sequence of m'; it then shows that with high probability there exists a milestone which lies in the intersection of the visibility sets of these two linking sequences, so that m and m' are in the same component of the roadmap.

Theorems 1 and 3 imply that with high probability, `roadmap` generates a roadmap that adequately represents F. Theorem 3 tells us that the probability that a roadmap does not capture F's connectivity decreases exponentially with the number of milestones, and the number of milestones needed increases moderately when ϵ, α, and β decrease.

Theorems 1, 2, and 3 do not explicitly mention the dimension n of $\mathcal{C}$. This comes from the fact that the definitions of ϵ-goodness and expansiveness only refer to volumes of subsets of F. But the dependence on n may be hidden in the parameters ϵ, α, and β. Consider the example of Figure 5. The free space consists of two cubes whose sides have length W; these cubes are connected by a narrow passage of length W and width w, with $w \ll W$. Up to a constant factor, each of the parameters ϵ, α, and β is on the order of w/W. The points with the smallest ϵ-goodness are located in the narrow passage. Each such point sees only a subset of F of volume approximately $3wW$; hence, $\epsilon \sim w/W$. A point near the top right corner of the left square sees this entire square; but only a subset of this square, of approximate volume wW, contains points that, each, see a set of volume $2wW$; hence, $\alpha \sim w/W$ and $\beta \approx w/W$. In the n-D version of this example, two hypercubes, each having volume W^n, are connected by a hyper-parallelepipedic passage that has size w along k dimensions ($k \in [1, n-1]$) and size W along the $n-k$ other dimensions. Each of the parameters ϵ, α, and β is on the order of $(w/W)^k$. The worst case happens when $k = n-1$, that is, when the passage is narrow along $n-1$ dimensions.

4.3 Path-Clearance Assumption

Another analysis of the basic PRM can be done by explicitly considering the "width" of the passages in F [13, 14]. Let q and q' be two configurations in the same component of F and τ be a free path connecting them. Let ℓ be the Euclidean length of τ and σ its distance to F's boundary. We call σ the *clearance* of the path.

Theorem 4. *Let ζ be a constant in $(0,1]$ and a be the constant $2^{-n}\mu(\mathcal{B}_1)/\mu(F)$ where $\mathcal{B}_1$ denotes the unit ball in $\mathbb{R}^n$. If s is chosen such that:*

$$s \geq \frac{1}{a\sigma^n} \log \frac{2\ell}{\sigma\zeta},$$

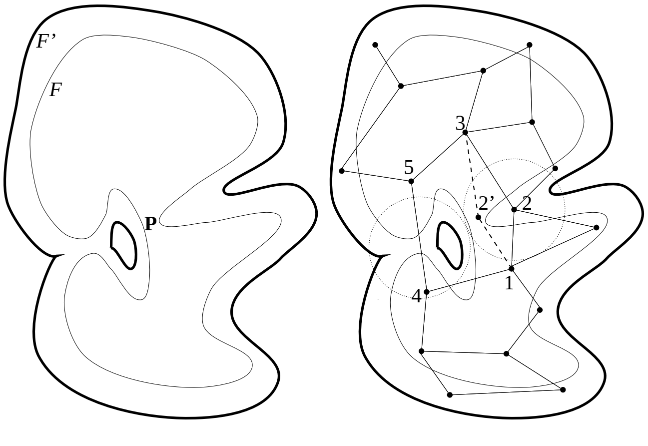
then with probability at least $1-\zeta$ `roadmap` *generates a roadmap in which one component contains two milestones m and m' such that q sees m and q' sees m'.*

This theorem is proven in [14]. Consistently with Theorem 3, it says that the probability that a roadmap fails to provide a path through a narrow passage decreases exponentially with the number of milestones. It also rightly suggests that the number of milestones may increase as $(2/\sigma)^n$. However, it is more conservative than Theorem 3. For instance, in the n-D version of the example shown in Figure 5, $\sigma = w/2$, even if the passage is narrow along only one dimension. While Theorem 4 suggests that the number of milestones needed increases as w^{-n}, Theorem 3 tells us that it only increases as w^{-k}, where $k < n$ is the number of dimensions along which the passage is narrow.

A variant of the path-clearance assumption that may yield a slightly tighter bound than Theorem 4 is the σ-complexity assumption proposed in [27].

5 Finding Narrow Passages

We first describe a new sampling strategy for finding narrow passages. Then we show experimental results with a simplified implementation of this strategy. The computation of penetration distances is studied in the next subsection. Finally, we discuss the new strategy in the context of the results of Section 4.



(a) (b)

Figure 6: *Roadmap construction steps*

5.1 New Algorithms

The new roadmap-construction algorithm, `new-roadmap`, starts by constructing a roadmap R' in a dilated free space F' obtained by allowing some penetration distance δ of the robot into the obstacles. Next, it "pushes" the milestones and links of R' that do not lie in F into F by performing local resampling operations. The outcome is a roadmap R in F. The rationale behind this two-stage strategy is that free space dilatation widens narrow passages and thus makes it easier to capture free space connectivity. Once constructed, R' provides pertinent information about which areas of F should be more densely sampled.

The operations of `new-roadmap` are illustrated in Figure 6. Figure 6(a) depicts the free space F (thin contour) and the dilated free space F' (bold contour). Note the narrow passage **P** in F. This passage is wider in F', but a spurious passage has been created on the left. Figure 6(b) displays a roadmap R' (only a subset of the links are shown) and illustrates resampling operations. The procedure `new-roadmap` resamples a neigborhood of 2, which is not in F, until it finds a new milestone $2'$ in F through which it can connect the milestones 1 and 3. Similarly, `new-roadmap` resamples a neigborhood of the link between milestones 4 and 5, but this operation eventually fails to find a connection. Other resampling operations are performed for milestones and links that are not in F, but are not illustrated in the figure.

procedure `new-roadmap`:

1. Generate a roadmap $R' = (M', L')$ of size s' in the dilated free space F' by invoking `roadmap`.
2. Initiate a roadmap $R = (M, L)$ in the free space F by setting M and L to the empty set.
3. **for each** milestone m in M' **do:**
   (a) **if** m lies in F **then** add m to M.
   (b) **else repeat** x **times:**
      i. Pick a configuration q uniformly at random in $U_m(m)$.
      ii. **if** q lies in F **then** add q to M.
      until a configuration q has been added to M.
   (c) **if** a configuration has been added to M at any of the above steps, **then** denote this configuration by $p(m)$.
4. **for each** link (m, m') in L' **do:**
   (a) **if** $p(m)$ sees $p(m')$ **then** add $(p(m), p(m'))$ to L.
   (b) **else**
      i. Pick y configurations uniformly at random in $U_l(p(m), p(m'))$. Let Q be the subset of those configurations which are in F.
      ii. **if there exists** a sequence $q_1, ..., q_k$ of configurations in Q, **such that** $p(m)$ sees q_1, q_i sees q_{i+1} for $i = 1, ..., k-1$, and q_k sees $p(m')$, **then:**
      . Add every q_i, $i = 1, ..., k$ to M.
      . Add every (q_i, q_{i+1}), $i = 0, ..., k-1$ to L.

Figure 7: *The new roadmap-construction algorithm*

Figure 7 gives a more formal description of `new-roadmap`. This procedure is given a function U_m that maps any configuration m to a region of fixed size and geometry around m, which constitutes the resampling region for pushing a milestone m into F. It is also given a function U_l that maps any pair of configurations (m, m') to the resampling region used for pushing a link (m, m') into F. Step 1 builds the roadmap R' in F', while Steps 3 and 4 push this roadmap into F. Step 3 pushes each milestone m of R' that is not already in F by picking up to x configurations at random in the region $U_m(m)$. The milestone of R obtained by pushing m is denoted by $p(m)$. If m was already in F, then $p(m) = m$. Step 4 pushes each link (m, m') of R' into F; this operation may lead to adding up to y milestones

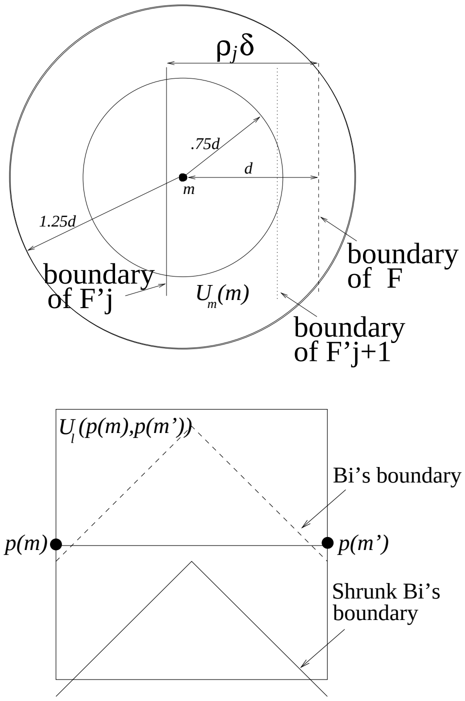


Figure 8: *The regions U_m and U_l*

to R by resampling the region $U_l(p(m), p(m'))$.

The new query-processing algorithm uses `query` to connect the two query configurations q_b and q_e to two milestones m_b and m_e of R'. For $i = b, e$, if q_i is directly connected to m_i, *i.e.*, q_i sees m_i in F', then, if needed, the new algorithm pushes the link $(q_i, p(m_i))$ into F by resampling the region $U_l(q_i, p(m_i))$. If, instead, q_i is connected to (*i.e.*, sees) an intermediate configuration $q \in F'$ that is itself connected to m_i, then q is first pushed to a configuration $p(q) \in F$ by resampling the region $U_m(q)$; next, both $(q_i, p(q))$ and $(p(q), p(m_i))$ are pushed into F by resampling the regions $U_l(q_i, p(q))$ and $U_l(p(q), p(m_i))$.

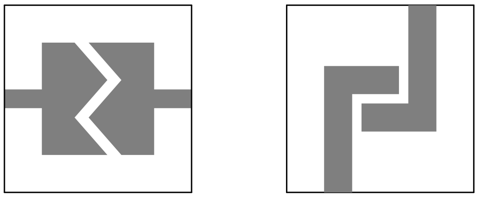
5.2 Implementation and Results

We have written two programs that respectively implement the algorithms `roadmap` and `new-roadmap`. These programs share large portions of code to facilitate comparison. They both apply to polygonal free spaces. Like the planner in [13, 16], our implementation of `roadmap` only tries to connect two milestones by a link if they are closer than some predefined distance. In all the experiments reported below this distance was set to 0.5. To investigate the behavior of the new algorithm in higher-dimensional spaces, a version of `new-roadmap` has also been adapted to the case where F is a polytope in 6-D.

In the 2-D case, the free space F is input by defining several obstacles B_i, $i = 1, 2, \ldots$, in $\mathcal{C}$. Our programs access this representation by invoking a procedure `cdist`(q, B_i) that returns the Euclidean distance between q and B_i's boundary. We use the notation "`cdist`" to stress the fact that the distance is computed in $\mathcal{C}$, which would not be the case in a more general implementation. This distance is positive if $q \notin B_i$, while it is negative if $q \in B_i$. Each obstacle B_i is a simple polygon, and no two obstacles B_i and B_j, $i \neq j$, intersect.

Our `new-roadmap` program uses a series of dilated free spaces, $F'_1, F'_2, \ldots, F'_r$, instead of a single one. The penetration distance for each F'_j $(j = 1, \ldots, r)$ is set equal to $\rho_j \delta$, where δ is the maximal penetration distance allowed and $1 \geq \rho_1 > \rho_2 > \ldots > \rho_r > 0$. In our experiments, r ranged between 1 and 5 and we set ρ_j to $(1/4)^{j-1}$ (hence, $\rho_1 = 1$). For convenience, we define F'_{r+1} to be F. The `new-roadmap` program constructs successive roadmaps $R'_1, \ldots, R'_r, R'_{r+1}$, where $R'_{r+1} = R$. Each milestone m of R'_j, $j \in \{1, \ldots, r\}$, that does not lie in F'_{j+1} is pushed into F'_{j+1} as follows. Let B_i be the obstacle in which m lies and let $d = -$`cdist`(m, B_i). Up to x configurations are picked at random in the circular ring $U_m(m)$ shown in Figure 8 (top), which is made of all points that are distant from m by more than $.75d$ (if $j \neq r$) or d (if $j = r$) and less than $1.25d$, until one configuration is picked in F'_{j+1}. Note that the definition of U_m is related to the choice of ρ_j. For each link (m, m') of R'_j, if the line segment joining $p(m)$ to $p(m')$ does not fully lie in F'_{j+1}, then up to y additional milestones are added in the square $U_l(p(m), p(m'))$ shown in Figure 8 (bottom). In all the experiments reported below we set x to 25 and y to 10. The choices of U_m and U_l reflect ad hoc compromises. In particular, the choice of U_l allows $p(m)$ and $p(m')$ to lie near the boundary of an obstacle B_i on both sides of a 90dg corner (dashed line in Figure 8).

We experimented with `roadmap` and `new-roadmap` on several examples with narrow passages. In our experiments we only worried about the connectedness of the final roadmap. Figure 9 shows two examples, where $\mathcal{C}$

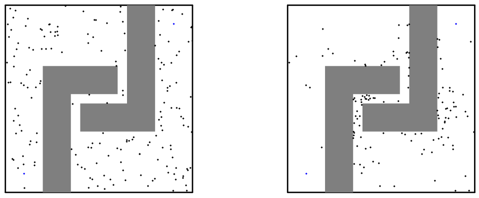


Example 1 Example 2

Figure 9: *Two examples*

Algorithm	Example 1 #mil.	Example 1 #cdist	Example 2 #mil.	Example 2 #cdist
`new-roadmap`	65	16,812	112	11,035
`roadmap`	281	67,894	290	70,971

Figure 10: *Experimental results*



`roadmap` `new-roadmap`

Figure 11: *Milestone distributions for Example 2*

w/W	r	#mil. (initial)	#mil. (final)	#cdist
.1	1	33	95	9,320
.01	1	38	118	32,942
.001	4	34	200	79,931
.0001	4	35	193	72,944
.00001	5	34	191	82,958

Figure 12: *Results with the example of Figure 5*

is the square $[0,1]^2$. In both examples, we set r to 1, and the maximal penetration distance δ to 0.075. In Example 1, this penetration distance causes the creation of new passages between obstacles. Each *test* with one program consisted of performing a sequence of runs. For the first run in a test of `new-roadmap`, the size s' of the roadmap $R' = R'_1$ is set large enough so that the final roadmap R includes a path through the narrow passage. The following runs in the test are done with iteratively smaller values of s', until the connectivity of the final roadmap no longer reflects that of F. We call the run just before this last run the *breaking run* of the test. Each test of `roadmap` is similar. The value of s is first set large enough that the roadmap R includes a path through the passage. The following runs are done with iteratively smaller values of s, until the connectivity of R is incomplete. We did 10 tests for each example and each program, using 10 different seeds for the random number generator; we used the same 10 seeds for both programs. The table in Figure 10 shows the average numbers of milestones in the roadmap R generated in the breaking runs and the average numbers of times `cdist` was invoked in those runs. The running time is not significative and is not indicated. In both examples we see that `new-roadmap` captures the free space connectivity with, on the average, considerably fewer milestones than `roadmap`; the average number of invocations of `cdist` is also much smaller. Figure 11 displays distributions of milestones generated for Example 2 by a breaking run of `roadmap` (left) and a breaking run of `new-roadmap` (right).

In another series of experiments with `new-roadmap` we considered the example of Figure 5, with w/W successively set to .1, .01, .001, .0001, and .00001. $\mathcal{C}$ was defined as $[0,3] \times [0,1]$. The maximal penetration distance was set to 0.45. For each value of w/W, we did the same kind of tests as above. The table of Figure 12 displays the average results over 5 tests. We used $r = 1$ for $w/W = .1$ and .01, $r = 4$ for $w/W = .001$ and .0001, and $r = 5$ for $w/W = .00001$. In this table, we indicate the numbers of milestones in the initial roadmap (R'_1) and in the final one (R). Note that the initial and final numbers of milestones, as well as the number of invocations of `cdist`, remain approximatively constant when w/W decreases from .001 to .00001. (The average running time also stayed roughly constant.) When $w/W = .00001$, the average number of milestones in the final roadmap is 191; in the same free space, `roadmap` would have had to generate on the order of 1000 times as many milestones, and we believe that this factor would continue to grow if w/W was set smaller. Figure 13 shows the distribution of milestones in a final roadmap in the case where $w/W = .001$ (the passage is barely visible in the figure). Due to the resampling at Step 3(b), this distribution is denser near the obstacle boundary. Due to the resampling of Step 4(b), it is maximum near the two entrances of the passage.

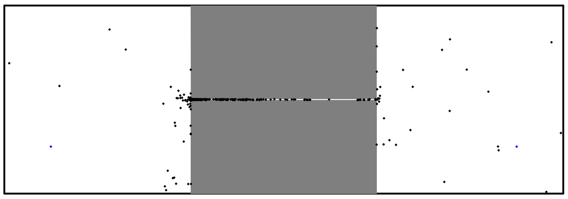


Figure 13: *Milestone distribution for example of Fig. 5*

k	average #mil. (new-roadmap)	estimated #mil. (roadmap)
1	61	40
2	105	800
3	1656	16,000

Figure 14: *Results with 6-D example*

We have adapted our `new-roadmap` program to handle the 6-D version of the example of Figure 5. We did experiments with $w/W = .05$ and $k = 1$, 2, and 3 (recall that k is the number of dimensions along which the passage is narrow). The second column of the table of Figure 14 shows the average numbers of milestones (over 5 runs) in roadmaps created by `new-roadmap` that are connected through the narrow passage. The third column contains an estimate of the numbers of milestones that `roadmap` would have to pick to build connected roadmaps with high probability.

5.3 Computation of Penetration Distance

The function `cdist` used in our implementation of `new-roadmap` computes the (penetration) distance in configuration space. But for path-planning problems involving objects of complex geometry and robots with many degrees of freedom, it is impractical to perform this computation in configuration space.

In Section 2 we introduced the function `dist`(q) that computes the Euclidean distance of the robot placed at q and the obstacles. To be more specific, let the robot be made of p rigid bodies $L_1, ..., L_p$. Let t_i be the shortest translation that L_i must undergo before it touches an obstacle. The distance `dist`(q) is $\min_{1=1}^{p}\{t_i\}$. As we mentioned in Section 2, a number of techniques have been proposed to implement `dist` [7, 8, 12, 18, 19, 20, 22, 25].

To be usable by `new-roadmap`, the function `dist`(q) should be extended to compute the (negated) penetration distance of the robot into the obstacles at configurations where $q \notin F$. Then, $q \in F'$ if and only if `dist`$(q) > -\delta$. Consistently with the above definition of `dist`(q) when $q \in F$, we can define the penetration distance of the robot placed at $q \notin F$ as follows. Let t_i be the shortest translation that L_i must undergo before it does not collide with any obstacle. The penetration distance `dist`(q) is $\max_{i=1}^{p}\{t_i\}$. Techniques for computing this penetration distance are proposed in [6, 7, 17, 26] for convex polytopes and in [20] for nonconvex ones. Note that this distance does not correspond to any metric in $\mathcal{C}$. Hence, allowing a penetration distance δ of the robot into the obstacles does not result in a uniform dilatation of F.

Another penetration distance is defined in [21, 22]. It is based on the notion of the "growth distance" between two objects and it is computed by growing/shrinking objects about a seedpoint. The computation of this distance has been implemented for convex obstacles.

To experiment with our sampling strategy on realistic path-planning problems, we project to implement a new version of the `new-roadmap` that uses a function `dist` based on V-Clip [20].

5.4 Discussion

We now discuss several aspects of the new sampling strategy in light of the results presented in Section 4. We recognize that a more formal analysis will eventually be desirable. For simplification, we assume that the dilated free space F' is obtained by growing F by δ isotropically. As mentioned above, allowing some penetration distance δ of the robot into the obstacles does not entail a uniform dilatation of F by δ. But, for any given robot, there exists a positive constant h such that $F \subset F_h \subseteq F'$, where F_h is obtained by uniformly dilating F by $h \times \delta$. We also assume that the volume of F' is not much greater than that of F, which simply means that δ is relatively small. This makes sense since the only thing we want is to significantly widen those passages which are too narrow to be found by `roadmap`.

• Consider the path-clearance assumption of Subsection 4.3. If two configurations q and q' are connected by a path of small clearance σ in F, the same path has clearance $\sigma + \delta$ in F'. This reduces the bound on the number of milestones given by Theorem 4 by a factor approximately equal to $1/(1+c)^n$, where $c = \delta/\sigma$.

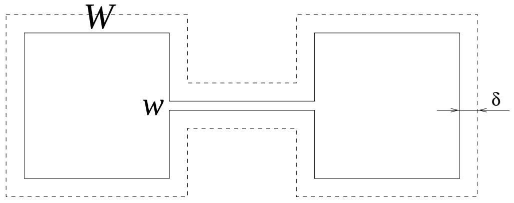
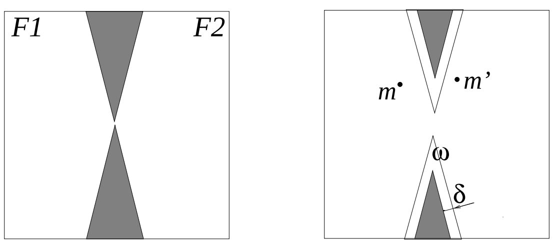


Figure 15: *Example of a dilated free space*



(a) (b)

Figure 16: *A pathological case*

In the example of Figure 5, c can be made very large. Hence, a relatively small roadmap R' can reliably capture the connectivity through the narrow passages of F.

• Consider the example of Figure 5. Recall that this problem has an n-D version where the passage is narrow along k dimensions, with $k = 1, 2, ...,$ or $n-1$. In 2-D, F' is the region depicted with a dotted boundary in Figure 15 (assuming that F' does not extend beyond $\mathcal{C}$'s boundary). In n-D, F is $(\epsilon, \alpha, \beta)$-expansive for values of ϵ, α, and β that are on the order of $(w/W)^k$. In F' the values of these parameters are on the order of $[(w+\delta)/W]^k$. So, posing $c = \delta/w$, the ϵ-goodness and expansiveness of F' are greater than those of F by a factor of $(1+c)^k$. The higher k, the greater this factor, which means that the worst case for F (which occurs when $k = n-1$) also yields the best improvement. Again, this means that a relatively small roadmap R' can reliably capture the connectivity of F.

The above two arguments indicate that, when a passage gets narrower, the gain in number of milestones achieved by `new-roadmap` over `roadmap` is on the same order of magnitude as the number of additional milestones that `roadmap` must generate to keep the roadmaps connected. This indication is consistent with the observation that the number of milestones remains approximately constant in Figure 12.

• Every milestone m picked at Step 1 in F', but not in F, is pushed into F at Step 3(b) by resampling $U_m(m)$. This operation produces a milestone distribution in F that is denser near its boundary. This effect is accentuated further by Step 4(b). If F has poor ϵ-goodness or expansiveness, the regions responsible for that are necessarily close to its boundary. Therefore, the resampling operations at Steps 3(b) and 4(b) increase the density of milestones where they are they are likely to be the most useful.

• In general, there is no guarantee that F' is more ϵ-good and/or expansive than F. However, we can define the ϵ-goodness and expansiveness of F through F' by extending the visibility relation as follows: two configurations in F see each other *through* F' if the straight line segment joining them lies in F'. The ϵ-goodness of F through F' is greater than its ϵ-goodness. Let S be a subset of a component E of F. Let the lookout of S through F' be the set of all points in S that see a significant portion of $E \backslash S$ through F'. The expansiveness of F through F' is greater than its expansiveness. Consider the roadmap R'' that contains all the milestones of R' lying in F and all the links between these milestones. (While all the milestones of R'' lie in F, its links may lie in F'.) Theorems 1 and 3 indicate that R'' can capture the connectivity of F (through F') with fewer milestones than a roadmap generated by `roadmap` in F.

• R' may contain more links than desired, since there may exist connections between milestones through F' that do not exist in F. In that case Step 4 will fail to push such links into F. It will waste some time trying to establish impossible connections. But it is reasonable to expect that this will only happen in a small number of localized areas. It is to be compared to the uniformly denser sampling that `roadmap` would have to perform to find existing narrow passages.

• The above comment raises the opposite question. Can Steps 3 and 4 of `new-roadmap` push the other links into F reliably? For example, shrinking the two triangular obstacles in Figure 16(a) by a relatively small δ causes a big widening of the passage, as depicted in Figure 16(b). The two milestones m and m' in this figure see each other through F', but pushing the link between them into F would require resampling a region $U_l(m, m')$ that would grow arbitrarily large as the angle ω decreases. The choice of U_l in our implementation (Figure 8) rests on an ad hoc compromise. When $\mathcal{C}$ has many dimensions and penetration distances are computed in the robot's workspace, there seems to exist no easy way to optimally set U_l's size. The use of successive dilated free spaces, as in our implementation, is one way to reduce the risks of disconnecting a roadmap when U_l is too small. Another way would be to try to connect pushed milestones $p(m)$ to more milestones in R. Finally, one should also note that a typical roadmap contains redundant links; hence, if Step 4 fails to push a link into F because U_l is too small, it may successfully push another nearby link and still get an adequate roadmap R.

• One may be tempted to choose a large value for δ in order to widen narrow passages as much as possible. However, a large value of δ may also cause large portions of obstacles or entire obstacles to vanish. At one extreme, if the dilated free space F' becomes the entire configuration space, the roadmap R' does not provide pertinent information to `new-roadmap` on which regions should be more densely sampled. To illustrate, consider Example 2 in Figure 9. One could make the two obstacles very "skinny" and the passage between them arbitrarily narrow. Setting δ to a value greater than half the thickness of the obstacles eliminates these obstacles and makes F' useless. Alternatively setting δ to a smaller value only results in a small widening of the passage, which hence remains hard to find.

• In Subsection 5.1 we could have kept the query-processing algorithm unchanged. However, it is possible that by reducing the number of milestones in R, these milestones may not adequately cover F, even though they may adequately cover F through F'. For that reason, the new query-processing algorithm connects the query configurations to R' and, if needed, pushes the connections into F.

6 Conclusion

This paper investigates the effect of narrow passages on the performance of a PRM. Section 4 provides foundations for understanding the effect of narrow passages on probabilistic roadmaps. It proposes three geometric properties, ϵ-goodness, expansiveness, and path-clearance, to assess the complexity of a path-planning problem submitted to a PRM. Section 5 uses these properties to propose and motivate a new approach to effectively deal with such passages. It also presents encouraging experimental results with an implementation of this approach in a polygonal configuration space.

It is clear, however, that much additional work is needed to formally analyze the proposed strategy, to understand the role of key parameters (e.g., δ, U_m, U_l, x, y) and how they could be adjusted on-line, to efficiently compute penetration distances, and to conduct more realistic experimentation. The computation of penetration distances is likely to be the key implementation issue when the robot and the obstacles have complex geometry, since it is inherently more expensive than the computation of separation distances.

Acknowledgments

This research was supported in part by ARO MURI grant DAAH04-96-1-007. D. Hsu is currently supported by a Microsoft Fellowship. L. Kavraki is partially supported by NSF CAREER Award IRI-970228. The authors also thank Leo Guibas, who first pointed to us that increasing milestone density near the free space boundary could help finding narrow passages.

References

[1] N. Amato, O.B. Bayazit, L.K. Dale, C. Jones, and D. Vallejo. OBPRM: An Obstacle-Based PRM for 3D Workspaces. *These proceedings*, 1998.

[2] N. Amato and Y. Wu. A Randomized Roadmap Method for Path and Manipulation Planning. *Proc. IEEE Int. Conf. on Rob. and Aut.*, Minneapolis, MN, pp. 113-120, 1996.

[3] B. Baginski. Local Motion Planning for Manipulators Based on Shrinking and Growing Geometry Models. *Proc. IEEE Int. Conf. on Rob. and Aut.*, Minneapolis, MN, pp. 3303-3308, 1996.

[4] J. Barraquand, L.E. Kavraki, J.C. Latombe, T.Y. Li, R. Motwani, and P. Raghavan. A Random Sampling Scheme for Path Planning. *Int. J. of Rob. Res.*, 16(6):759-774, 1997.

[5] J. Barraquand and J.C. Latombe. Robot Motion Planning: A Distributed Representation Approach. *Int. J. of Rob. Res.*, 10(6):628-649, 1991.

[6] S.A. Cameron. Enhancing GJK: Computing Minimum and Penetration Distances Between Convex

Polyhedra. *Proc. IEEE Int. Conf. on Rob. and Aut.*, Albuquerque, NM,pp. 3112-3117, 1997.

[7] D.P. Dobkin, J. Hershberger, D.G. Kirkpatrick, and S. Suri. Computing the Intersection Depth of Polyhedra. *Algorithmica*, 9:518-533, 1993.

[8] E.G. Gilbert, D.W.Johhson, and S.S. Keerthi. A Fast Procedure for Computing the Distance Between Complex Robots in Three-Dimensional Space. *IEEE Tr. Rob. and Aut.*, 4:193-203, 1988.

[9] P. Ferbach and J. Barraquand. A Method of Progressive Constraints for Manipulation Planning. *IEEE Tr. Rob. and Aut.*, 13(4):473-485, 1997.

[10] D. Hsu, J.C. Latombe, and R. Motwani. Path Planning in Expansive Configuration Spaces. Proc. *IEEE Int. Conf. pn Rob. and Aut.*, Albuquerque, NM, pp. 2719-2726, 1997. An extended version of this paper will appear in *Int. J. of Comp. Geom. and Applicatons.*

[11] T. Horsch, F. Schwarz, and H. Tolle. Motion Planning for Many Degrees of Freedom - Random Reflections at C-Space Obstacles. *Proc. IEEE Int. Conf. on Rob. and Aut.*, San Diego, CA, pp. 3318-3323, 1994.

[12] P. Jiménez, F. Thomas, and C. Torras. Collision Detection Algorithms for Motion Planning. *Robot Motion Planning and Control*, J.P. Laumond (ed.), Lecture Notes in Control and Info. Sc., 229, Springer, New York, NY, pp. 305-343, 1998.

[13] L. Kavraki. *Random Networks in Configuration Space for Fast Path Planning.* Ph.D. Thesis, Rep. No. STAN-CS-TR-95-1535, Comp. Sc. Dept., Stanford Univ., Stanford, CA, 1995.

[14] L. Kavraki, M. Kolountzakis, and J.C. Latombe. Analysis of Probabilistic Roadmaps for Path Planning. *Proc. IEEE Int. Conf. on Rob. and Aut.*, Minneapolis, MN, pp. 3020-3025, 1996.

[15] L. Kavraki, J.C. Latombe, R. Motwani, and P. Raghavan. Randomized Query Processing in Robot Motion Planning. *Proc. ACM SIGACT Symp. on Theory of Computing (STOC)*, Las Vegas, NV, pp. 353-362, 1995.

[16] L. Kavraki, P. Švestka, J.C. Latombe, and M. Overmars Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. *IEEE Tr. Rob. and Aut.*, 12(4):566-580, 1996.

[17] S.S. Keerthi and K. Shidharan. Measures of Intensity of Collision Between Convex Objects and Their Efficient Computation. *Proc. of Int. Conf. on Intelligent Robotics*, pp. 266-272, 1991.

[18] M. Lin and J.F. Canny. A Fast Algorithm for Incremental Distance Computation. *Proc. of the IEEE Int. Conf. on Rob. and Aut.*, Sacramento, CA, pp. 602-608, 1994.

[19] M. Lin, D. Manocha, J. Cohen, and S. Gottschalk. Collision detection: Algorithms and applications. *Algorithmic Foundations of Robotics*, Goldberg et al. (Eds), A K Peters, Ltd., pp. 129-141, 1995.

[20] B. Mirtich. *V-Clip: Fast and Robust Polyhedral Collision Detection.* Tech. Rep. TR97-05, Mitsubishi El. Res. Lab., Cambridge, MA, 1997.

[21] C.J. Ong. On the Quantification of Penetration between General Objects. *Int. J. of Rob. Res.*, 16(3):400-409, 1997.

[22] C.J. Ong and E.G. Gilbert. Growth Distances: New Measures for Object Separation and Penetration. *IEEE Tr. Rob. and Aut.*, 12(6):888-903, 1996.

[23] M. Overmars. *A random Approach to Motion Planning.* Tech. Rep. RUU-CS-92-32, Comp. Sc. Dept., TB Utrecht, The Netherlands, 1992.

[24] M. Overmars and P. Švestka. A Probabilistic Learning Approach to Motion Planning. *Algorithmic Foundations of Robotics*, K. Goldberg et al. (eds.), A.K. Peters, Wellesley, MA, pp. 19-37. 1995.

[25] S. Quinlan. Efficient Distance Computation Between Non-Convex Objects. *Proc. IEEE Int. Conf. on Rob. and Aut.*, San Diego, CA, pp. 3324-3330, 1994.

[26] K. Sridharan, H.E. Stephanou, K.C. Craig, and S.S. Keerthi. Distance Measures on Intersecting Objects and their Applications. *Inf. Proc. Letters*, 51:181-188, 1994.

[27] P. Švestka and M. Overmars. Probabilistic Path Planning. *Robot Motion Planning and Control*, J.P. Laumond (ed.), Lecture Notes in Control and Information Sciences, 229, Springer, New York, NY, pp. 255-304, 1998.