5
Probabilistic Roadmap

So far, we have discussed various techniques including exact solutions (e.g., silhouette technique). Unfortunately, such a exact approach has a prohibitive time complexity as a function of dimensions, while various heuristic approaches like potential fields cannot guarantee to find a solution. Given this trade-off space, probabilistic roadmap arises a practical alternative, which can find a solution with a higher probability as we explore more on the C-space.

In this chapter, we discuss main components of Probabilistic RoadMaps (PRMs). PRM techniques assume that the environment is static and many queries of asking to find paths from multiple sources to multiple destinations. A common example of such cases is to use industrial manipulator that is fixed in a workcell (Fig. 5.1. Intuitively speaking, you can think of results of the PRM techniques as a common map that we use for finding paths.

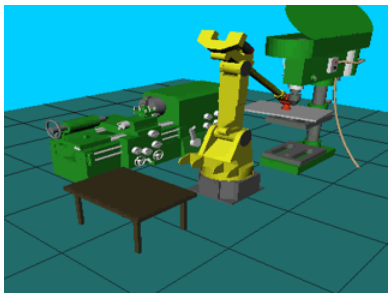

Figure 5.1: This shows an industrial manipulator fixed within a workcell. Its environment can be considered as static.

5.1 Overall process

At a high level, PRM has two main components: roadmap construction, i.e., a pre-computation of generating a map, and using the computed roadmap for processing various queries.

For computing the roadmap, the PRM approach generates random samples, q, on the C-space and check whether we do have collisions or not at those random samples q; see Fig. 5.2. PRM (Probabilistic roadmap) has its name, since we use such random samples. When there is no collision at the random sample q, we call it milestone and generate a node in the graph that serves as a roadmap; this operation is done by calling the Clear (q) operator, defined in Ch. 4.

For each free sample q, we identify nearby nodes q' and check whether we do not have any collisions on the straight line between q and q'; this is also done by calling the Link (q, q') operator, mentioned in Ch. 4. For the case of the collision-free line segment, we add it as an edge into the roadmap graph. We continue this process of

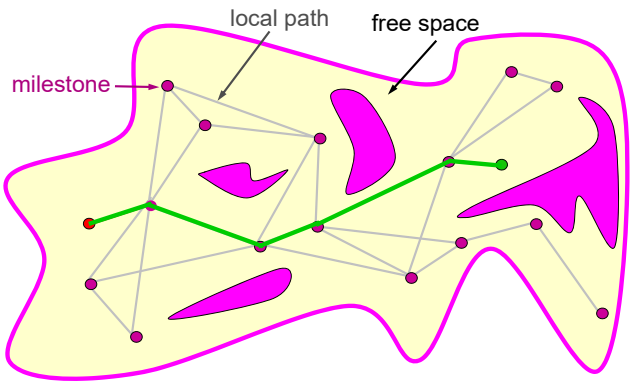

Figure 5.2: This shows a roadmap, i.e., graph, consisting of nodes (known as milestone) and edges (known as local path) which are located in the C-free space. Given start and goal, we connect them into nodes of the graph and find a path using the roadmap.

Algorithm 1: A basic construction method for probabilistic roadmap (PRM).

```
Data: Geometry of the robot and obstacles
Result: Roadmap, G = (V, E)
Initialize G
while Within the available time budget do
    q ← a randomly generated sample in the C-space
    if Clear (q) then
        Add q to G
        N_q ← a set of nearby nodes in V to q
        for each q' in N_q in an increasing distance do
            if Link (q, q') then
                Add an edge (q, q') to G
            end
        end
    end
end
```

generating random samples and connecting them with other nodes in the graph until a particular termination condition, e.g., computation budget. Its pseudocode is shown in Alg. 1.

One can see that as we generate more random samples, the probabilistic roadmap is getting larger with more connections among nodes, resulting in covering a wider set of homotopic classes in a probabilistic way. While we assumed a uniform sampling on generating samples, we can use different sampling methods that can achieve better quality in terms of covering the free space.

A common problem of using a naive sampling including the uniform sampling is that the graph may have many disconnected sub-graphs, which are not useful for finding paths. A simple way of constructing better connected graph is to use an expansion technique

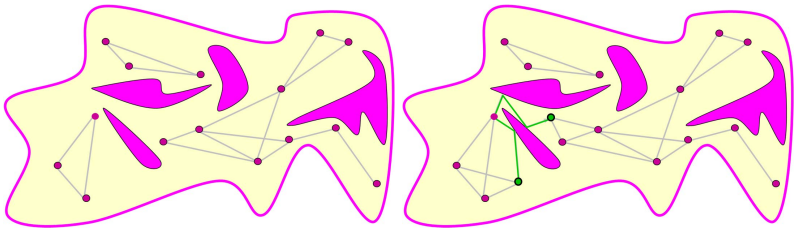

Figure 5.3: A naive sampling may result in many disconnected sub-graphs as shown in the left. By using a better connection method, we can connect those sub-graphs as shown in the right.

as shown in the right side of Fig. 5.3. Its idea is to identify a set of nodes that are in difficult regions in terms of connection with other nodes. To know such nodes, we can simply measure how many times we fail to connect them with other nodes. For nodes with a high probability of connection failure, we walk on a random direction in the C-space until we hit an obstacle. When we hit an obstacle, we move along another random direction, and iterate this process. During this expansion process, we wish to get out of such difficult regions, e.g., narrow passage. We then check whether we can have a connection between the current position and other nearby nodes.

The main technical question is how to prove the probabilistic completeness of a given PRM method. This is discussed in Sec. 5.3.

Path smoothing. A path computed from PRM methods can be very jerky, since the computed paths are a sequence of edges in the probabilistic roadmap. Such jerky paths are not desirable ones for mobile robots to track. It is a common to smooth such paths as a post-processing (Fig. 5.4).

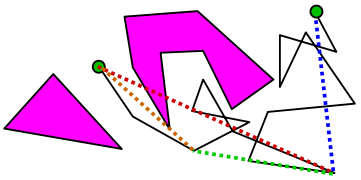

Figure 5.4: We commonly smooth the jerky path computed from PRM as a post-processing.

A simple smoothing method is to test two simple segments, one of which connects the start and the middle node in the initial path, and another of which connects the middle node and the goal. When each of simple segments does not have any collision, we simply use them as a smooth path. When a segment is in C-obstacle space, we perform the same operation on that segment in a recursive manner.

5.2 *Sampling Strategies*

In this section, we discuss a few sampling strategies to design more effective roadmaps than the aforementioned, basic one.

Let us first consider the visibility based PRM method [1]. Its main idea is that when we can see a node q_i from another node q_j, we may not need to encode the node q_i as a milestone in the roadmap, resulting in a more compact representation.

This approach uses two types of nodes, guard and connection

[1] T. Simeon, J. P. Laumond, and C. Nissoux. Visibility based probabilistic roadmaps for motion planning. *Advanced Robotics Journal*, 14(6), 2000

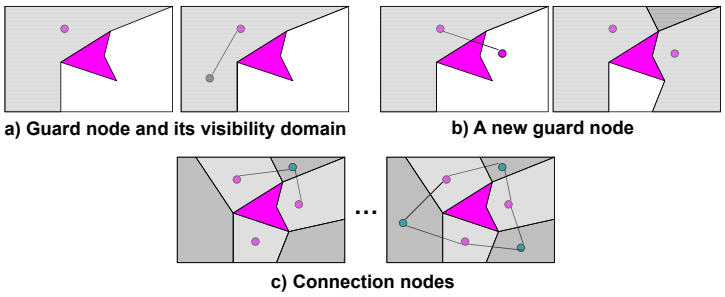

Figure 5.5: This shows guard and connection nodes of the visibility based PRM.

nodes. Guard nodes are constructed in a way that any two guard nodes cannot see each other, serving as a compact node set for the roadmap. Only with these guard nodes, we cannot connect those nodes. We thus use connection nodes, each of which can be seen from two guard nodes. Once we identify such connection nodes, we connect those two guard nodes through edges between the connection node and its associated guard nodes.

To identify whether a randomly generated node can be a guard node or not, we also utilize the concept of visibility domain. Suppose that we have a guard node as shown in the purple node of a) of Fig. 5.5. Its visibility domain contains any points that can be seen through a straight line or a local planner from the guard node. We then generate a new node and check whether it can be a new guard node or not. In the right figure of a), the grey node is visible from the existing guard node and we thus reject it. For another purple node of Fig. 5.5-b), we cannot see it from the existing guard node and thus we treat it as a new guard node. In this way, we can compute guard nodes and increasing the coverage of the free space.

Fig. 5.5-c) shows how we compute connection nodes. When a random node (e.g., dark green nodes) is visible from two guard nodes, we connect them with edges.

Once we construct the roadmap, called visibility roadmap, with guard and connection nodes, we can compute a path from start and goal position, by finding two guard nodes that see those start and goal positions and edges connecting those guard nodes.

YOON: Discuss Guassian sampling method

5.3 *Probabilistic Completeness*

In this analysis, we focus a simplified PRM, s-PRM, which does not use any heuristic nor shortcut to achieve better performance. In s-PRM, we generate N different random points in D dimensional C-space, and then connect any of those two points that can be connected by a free straight line.

Theorem 5.3.1. *We assume that there is a feasible path, ρ, and let R to be the smallest distance between the path and obstacles. When generating N random samples, the probability of failing to find the path whose length is L is defined as the following* [2]*:*

$$P(Fail) \leq \frac{2L}{R}(1 - \alpha_D R^D)^N, \tag{5.1}$$

where $\alpha_D = 2^{-D}\frac{\pi^{D/2}}{\Gamma(D/2+1)Vol(C_{free})}$, $Vol(\cdot)$ *is the volume of the given space, and* $\Gamma(\cdot)$ *is the gamma function.*

[2] E.E. Kavraki, M.N. Kolountzakis, and J.-C. Latombe. Analysis of probabilistic roadmaps for path planning. *Robotics and Automation, IEEE Transactions on*, 14 (1):166–171, 1998

Proof. Let us to use $B_R(x_i)$ to denote the unit ball with a radius R whose center is at x_i. To provide the probability, we take the following steps:

1. We generate balls covering the path such that when we can generate at least a single random point in each ball, we can construct a path that is homotopic to the feasible path.

2. We then compute a probability that we cannot generate any samples on those balls.

In the first step, we generate n different balls with $R/2$ such that those balls cover the path ρ. To do that, n is set to be $\frac{L}{R/2} = \frac{2L}{R}$, and those balls are located at $x_0 = \rho(0), x_1 \cdots, x_n = \rho(L)$, where $d(x_j, x_{j+1}) \leq R/2$ for all j. In this configuration of balls, we have the following equation:

$$B_{R/2}(x_{j+1}) \subseteq B_R(x_j), \text{for appropriate } j. \tag{5.2}$$

As a result, once we have random samples on those two balls j and $j+1$, the edge between those samples are located at $B_R(x_j)$ and does not have any collisions against obstacles (Fig. 5.6). Also, in this ball, the path segment can be transformed to the feasible path.

As the second step, we now look at a probability of not generating a random sample at one of those balls. Suppose that $q_1, \cdots, q_N$ are random samples generated by our planner. If we can generate those samples at least one at each ball, we construct a path that is homotopic to the feasible path. The probability that a ball, $B_{R/2}(x_j)$, does not contain any of them is $(1 - Vol(B_{R/2})/Vol(C_{free}))^N$. The

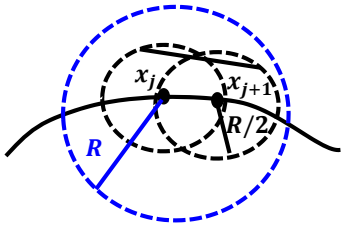

Figure 5.6: A configuration for error analysis.

probability that we want to compute is defined as the following:

$$\begin{aligned} P(Fail) &< P(\text{Some balls are empty}) \\ &< \sum_{j=1}^{n-1} P(\text{j-th ball is empty}) \\ &< \frac{2L}{R}(1 - \frac{Vol(B_{R/2})}{Vol(C_{free})})^N \\ &= \frac{2L}{R}(1 - 2^{-D}\frac{Vol(B_R)}{Vol(C_{free})})^N \\ &= \frac{2L}{R}(1 - 2^{-D}\frac{\pi^{D/2}R^D}{\Gamma(D/2+1)Vol(C_{free})})^N. \end{aligned} \tag{5.3}$$

□

Simplified representation and its characteristics. Using the inequality, $1 - x \leq e^{-x}$, for $x \geq 0$, the bound of Theorem 5.3.1 becomes:

$$P(Fail) \leq \frac{2L}{R} exp(-\alpha_D R^D N). \tag{5.4}$$

The bound drops exponentially on the number of random samples, and increases linearly to the path length. In this derivation, we use the smallest R to the distance between the path and obstacles, but we can also derive a similar bound by using variable R_t as a parameter t in the path [3].

It has been shown that PRM with heuristics such as k-nearest neighbor sPRM are not probabilistically complete [4], while RRT is as shown in Ch. **??** YOON: Fix .

[3] E.E. Kavraki, M.N. Kolountzakis, and J.-C. Latombe. Analysis of probabilistic roadmaps for path planning. *Robotics and Automation, IEEE Transactions on*, 14 (1):166–171, 1998

[4] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *Int'l. Journal of Robotics Research*, 30(7):846–894, 2011