Result

planner	Avg planning times	Success rate (<5s)	Avg number of vertices generated	Average Path Qualities (cost)
RRT	0.0137	100%	7.273	7.638
RRT Connect	0.0125	100%	8.364	13.831
RRT Star	0.0163	100%	8.483	6.386
PRM	2.8288	100%	3.262	4.344

- RRT creates a tree by randomly sampling points in space and connecting them incrementally to explore the environment. This works efficiently but does not create the smoothest path. Therefore, RRT takes a relatively short time with a higher number of vertices, compared to non-RRT method, here.
- RRT Connect has the same mechanism as RRT but grows two trees (from start and from goal) then connects them. This makes planning slightly faster than RRT in some environments but at the cost of path optimality. Therefore, my RRT Connect takes less time than RRT with higher cost.
- RRT star is an optimized RRT that focuses on reducing costs, finding a shorter and smoother path. Therefore, it makes sense that my RRT star has less cost than RRT and RRT Connect.
- PRM samples point in the environment upfront to form a roadmap that can be used repeatedly. The planning part is slower than other planners, but PRM can generate high quality paths, especially for known environments. Therefore, my PRM has the highest planning time and the lowest cost.

Since this is a known environment with no obstacle changes, PRM is the best since the samples points can be used repeatedly, the least number of vertices is needed, and it demands the least cost.

In a highly complex environment, the PRM would require significant upfront computation to build the roadmap. Sampling too many points could be expensive and inefficient.

To solve this problem and improve PRM, we can perhaps do adaptive sampling to strategically generate samples in area with more obstacles or area closer to start/goal points.

Extra credit:

I have compiled 10 sets of data with similar start and goal points.

Here is the variance that I calculated

planner	**solution_variance**
RRT	0.45
RRT Connect	0.52
RRT Star	0.25
PRM	0.12

This makes sense because PRM relies on a pre-computed roadmap. Since this is a set environment that does not change, the solution doesn't change much as well.

RRT and RRT connect both samples randomly and create very different paths every run.

RRT* focuses on finding the smoothest path and would have very consistent solutions for similar start and end goals.

High level detail

RRT: I utilized nearest neighbor search to find the closest existing node to a new random configuration, then attempt to expand toward it with a step size. To increase efficiency, I included a goal bias – within a small probability, the random sample is set to the goal configuration. While this bias doesn't guarantee reaching the goal, this allows the tree to grow toward the goal faster. Since the chance to reach the exact goal is small, I included a termination that allows the path to be found within specific step size.

RRT Connect: I included two trees, from start and from goal, that aim to connect at the middle. The trees grow alternatively. After expanding the start tree toward a random configuration, the algorithm finds the nearest node in the goal tree and attempts to grow it toward the new start tree node to speed up connection attempt.

RRT* : As an improvement of the RRT, this planner examines neighboring nodes after adding a new node to see if the new node offers a lower-cost path. If so, it updates those connections. Neighboring nodes within a specific radius are considered for wiring.

PRM: This planner samples random configurations, validates them, and connects valid configurations that are within a certain connection radius to create a roadmap graph. After the graph is constructed, PRM uses A* search to find the shortest path from start to goal within this roadmap.