**Algorithmic Robotics and Motion Planning – Final Project  
Dor Falah**

Motion planning is a complex problem in high-dimensional spaces where multiple robots must navigate together. During the semester exercises we solved that issue in low dimensions.  
This project focuses on extending the problem to higher dimension - up to five robots and implementing + comparing three motion planning algorithms: Probabilistic Roadmap (PRM), Rapidly exploring Random Tree (RRT), and RRT\* (RRT Star). Each algorithm is implemented and tested across various scenarios.

**Project Overview**

The primary objective of this project is to assess the performance of each algorithm in terms of two key metrics: the distance cost (Euclidean distance) of the paths (with shorter paths being preferable) and the success rate of finding a valid path (measured between 0 and 1). The analysis provides insights into the strengths and weaknesses of each algorithm and offers recommendations for their use in different robotic applications.

**Implemented Methods**

1. **Probabilistic Roadmap (PRM)**
   * PRM is the simplest of the three methods implemented. It constructs a roadmap of the environment by randomly sampling points and connecting them to form a graph of possible paths. PRM is the fastest due to its simplicity, but it often struggles with consistency in complex environments.
2. **Rapidly exploring Random Tree (RRT)**
   * RRT builds a tree by incrementally expanding paths from a start point towards the goal, favoring unexplored regions of the scene. This method is effective in finding paths but may not always produce the shortest routes.
3. **RRT\***
   * RRT\* enhances RRT by optimizing the paths to ensure they are as short as possible. For this project, I implemented the RRT\* algorithm with some modifications to improve runtime without compromising the path quality:
     + **Initial Full Execution**: For the first 2000 landmarks, the **full** RRT\* steps are executed to establish a strong foundation of optimized paths.
     + **Randomized XNEAR Group**: After 2000 landmarks, the XNEAR group is generated by shuffling the “so far” graph nodes and checking only a logarithmic number of nodes of it, significantly improving runtime while maintaining path quality.
     + **Periodic Full Iterations**: To ensure continuous optimization, a full iteration (generating full XNEAR group) of the RRT\* algorithm is performed every 500 nodes. This approach balances runtime efficiency with the need for near-optimal path lengths.
   * **Despite these improvements, RRT\* is the slowest algorithm among the three due to its complexity, making it the most time-consuming to execute.**

**Performance Evaluation Experiment**

In this experiment, I compared these algorithms performance across various scenes (7) and parameters. For each scene, I executed each algorithm with three different numbers of landmarks, performing 10 runs for each configuration (**630 runs total**). This approach provided a comprehensive understanding of the algorithms’ performance. I collected and averaged the results, and there are the findings:   
**\*At the end of this report, I’ve attached two graphs to visualize the results.**

**PRM (Probabilistic Roadmap)**

The PRM algorithm's performance is characterized by moderate to high average distances and variable success rates across different scenarios. Specifically:

* **3 Discs Custom**: PRM showed an average distance of 74.86 to 86.10 units and a success rate between 0.2 to 0.9 across different landmark counts.
* **4 Robots**: Here, PRM's performance improved with an increase in landmarks, but the success rate remained inconsistent, ranging from 0.1 to 0.5.
* **Hard Test 2 Robots**: The success rate varied from 0.2 to 0.7, with average distances generally higher than those achieved by RRT\*.
* **5 Robots**: PRM struggled significantly, achieving an average distance of up to 321.84 units and very low success rates.
* **3 Robots Hard Maze**: PRM failed completely, indicating a need for better handling of complex environments.
* **5 Robots Hard:** PRM consistently fails to find a valid path in this complex environment, with a success rate of 0.0 across all configurations.

Overall, PRM's utility appears **limited in more complex scenarios** due to its inconsistent success rates and relatively longer paths.

**RRT (Rapidly exploring Random Tree)**

RRT consistently achieved high success rates but at the cost of higher average distances compared to RRT\*. Specific observations include:

* **3 Discs Custom**: RRT maintained a success rate of 1.0 across all landmark counts but had average distances as high as 280.53 units.
* **4 Robots**: RRT again showed a high success rate (up to 1.0) but with average distances ranging from 243.01 to 475.52 units.
* **Hard Test 2 Robots**: RRT achieved perfect success rates but with average distances consistently above 133.41 units.
* **5 Robots**: The success rate was high (up to 1.0), but average distances were among the highest recorded, reaching 475.52 units.
* **3 Robots Hard Maze**: Despite a success rate of 0.7, the average distance was significantly higher at 491.14 units.
* **5 Robots Hard:** RRT shows improvement with more landmarks, increasing success rates from 0.3 to 0.6, but provides longer paths compared to RRT\*.

RRT is **highly reliable** in finding a path but **always** results in longer paths compared to RRT\*.

**RRT\***

RRT\* demonstrates **superior performance**, it achieved the average **shortest paths in all scenes** and performed the **highest success rates together with RRT**, making it the most efficient algorithm overall. Key highlights include:

* **3 Discs Custom**: RRT\* achieved the lowest average distances (49.48 to 50.70 units) with a consistent success rate of 1.0.
* **4 Robots**: The average distances ranged from 54.04 to 81.11 units, with success rates improving up to 1.0.
* **Hard Test 2 Robots**: RRT\* consistently outperformed other algorithms with average distances as low as 15.68 units and success rates of 1.0.
* **5 Robots**: Even in complex scenarios, RRT\* maintained average distances between 67.27 to 73.45 units and a high success rate.
* **3 Robots Hard Maze**: Despite the complexity, RRT\* achieved average distances of around 165.78 units with a success rate of 0.7.
* **5 Robots Hard:** Despite being the most complex scene,RRT\* provides the best success rate and provides the shortest paths with an average distance of 154.64.

**Comparative Analysis**

The comparative analysis highlights RRT\*'s substantial advantages in terms of path optimization:

**Average Distance**

* The average improvement of RRT\* over PRM in terms of shortest path ranges from around **10% to 79%**, depending on the complexity of the scene and the number of landmarks.
* In more complex scenarios, such as "5 Robots" and "Hard Test 2 Robots," the improvements are generally higher, **often exceeding 40%**.
* For simpler scenarios or those with fewer landmarks, the improvement is still significant but relatively lower, usually around **10% to 30%**.

**Success Rate**

* PRM shows significant variability in success rates, particularly struggling in more complex scenes (**0% success in “5 Robots Hard” and “3 Robots Hard Maze”**).  
  Its success rates range from 0 to 0.9, indicating that it **may not always be dependable, especially with fewer landmarks**.
* RRT maintains a high success rate, typically between 0.9 and 1.0 in most environments. **While it might not produce the shortest paths**, its high success rate ensures that it consistently finds a valid path, making it a reliable option for pathfinding tasks.
* RRT\* **consistently achieves high success rates**, often reaching 1.0 in various scenarios. This reliability makes it an **excellent choice for complex environments** where ensuring a valid path is crucial.

**Conclusions**

RRT\* emerges as the superior algorithm for robot motion planning among the three, offering the shortest paths and high success rates across various scenarios. Despite being the slowest algorithm due to its complexity, RRT\* consistently provides the most optimized paths, making it the recommended choice for applications where path efficiency is critical.

While RRT is also reliable in finding a path, it often results in longer distances. RRT is faster than RRT\* but does not achieve the same level of path optimization.

PRM, despite its simplicity and being the fastest of the three, struggles in complex environments and provides less optimal paths. Its speed comes at the cost of reliability and path quality, making it less suitable for scenarios requiring precise and efficient navigation.

In summary, for applications prioritizing path efficiency, RRT\* is the best choice despite its slower performance. RRT offers a balance between speed and reliability, while PRM is suitable only for simpler, less demanding environments.

**Results Visualization**

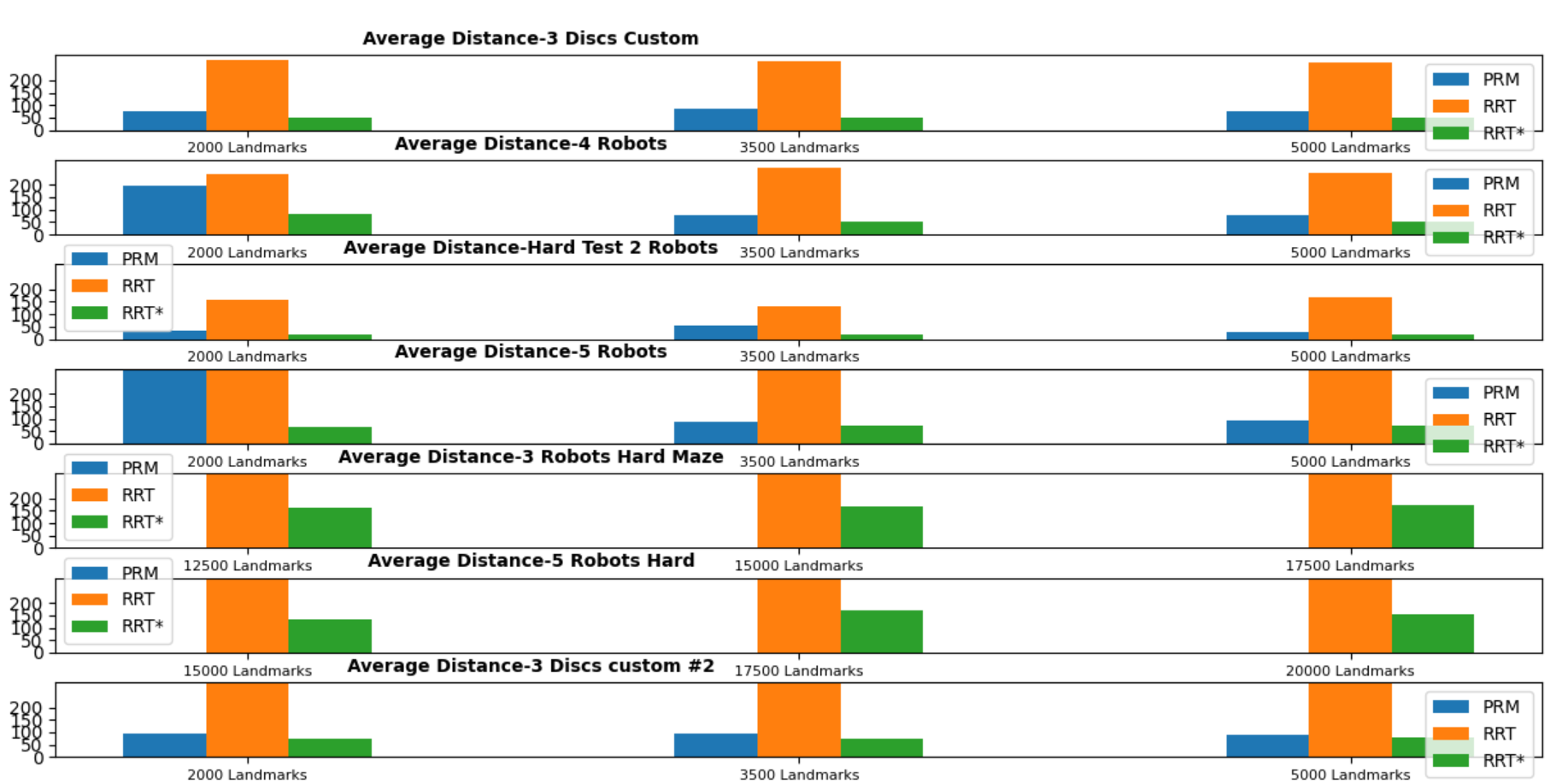
**Green-RRT\*  
Brown-RRT  
Blue-PRM**

**Success Rate (The higher the better)**

**A graph with different colored squares

Description automatically generated**

**Average Distance (The lower the better)**

****