Trajectory-Tracking Motion Planning for Mobile Manipulator System

Gokul M K(ED21B026) Keerthi Vasan M(ED21B034)

September 23, 2025

Videolink : *ClickHere*

Githblink : *Clickhere*

Abstract

This project is centered on following a specified trajectory in the task space of varying shapes using the end effector of the Kuka Youbot, while navigating around obstacles. We introduce a novel inverse kinematics solution method to convert task space coordinates into configuration space coordinates. By employing RRT* or BiRRT* in the 8-degree-of-freedom (DOF) system, we plan the trajectory efficiently. The outcomes illustrate that our algorithm adeptly tracks the pattern utilizing near-optimal sampling-based algorithms like RRT* and BiRRT*, both in the presence and absence of obstacles. Our methodology brings notable advancements in inverse kinematics determination for 8 DOF systems, transforming the conventional control tracking challenge into a motion planning problem formulation that remains effective even in obstacle-laden scenarios, unlike traditional control approaches.

1 Introduction

Mobile manipulators are increasingly being deployed across diverse sectors, ranging from warehouse automation to manufacturing. They play a crucial role in tasks like 3-D printing, balancing, and wall-painting, especially in large workspaces. However, these applications come with notable challenges, such as effectively managing redundancy within the kinematic chain and ensuring seamless trajectory continuity.

This project is focused on addressing such complex scenarios where the workspace includes obstacles, requiring autonomous movement and pattern tracking. To achieve this, a novel Inverse Kinematics planner is utilized to convert task space into configuration space, coupled with a sampling-based strategy for planning in an 8-degree-of-freedom (DOF) setting. Various algorithms like Rapidly Exploring Random Trees (RRT), RRT*, Bi-Directional RRT (BiRRT), and BiRRT* have been tested, with the selection of the two most effective ones.

Furthermore, Collision Avoidance is integrated into the Motion Planning process to chart a secure path to the target location. This involves thorough examination of multiple sample points within the environment before finalizing the plan for the Robot's movement.

Ultimately, the comprehensive integration of Sampling-Based Planners, Combined Inverse Kinematics, and Collision Avoidance is put to the test in a simplified environment, demonstrating the cohesive functionality of these components in a unified system.

2 Methods

2.1 Proposed Solution

Our proposed solution is broken down into 3 subparts: (1) Inverse Kinematics (2) Motion Planning (3) Collision Avoidance. From the given task space coordinates for our end effector. We first find the corresponding Configuration space coordinates using the below formulation:

2.1.1 Inverse Kinematics

Planning the inverse Kinematics for the base is easy since we know the corresponding task space coordinates and location of static obstacles. One simple way to think of this for a holonomic system is to go to the locus of radius R= 0.2 depending on the reachability of the arm and then plan the inverse kinematics for the 5 DOF system. But since the joints number is less than 6, it may or may not have a solution.

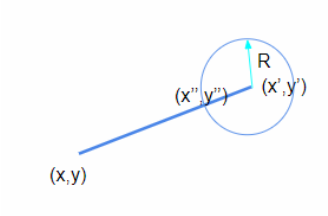


Figure 1: Simple IK for base

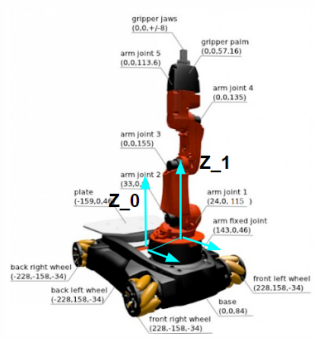


Figure 2: New Frame assignment

To solve this issue, we are introducing the **extended DH parameters for the arm with a pseudo joint at the base footprint formulation**. We assume a pseudo hip joint at the base footprint and extend it to a 6-DOF manipulator. This approximation is valid since the system is holonomic and we can control the base velocities individually. The modelled 6DOF manipulator system's DH parameters are shown below:

Table 1: Extended DH Parameter table

Link	a_{i-1}	α_{i-1}	d_i	θ_i
1	0.143	0	0.046	θ_1
2	0.033	$\frac{\pi}{2}$	0.147	θ_2
3	0.155	0	0	θ_3
4	0.135	0	0	θ_4
5	0	$\frac{\pi}{2}$	0	θ_5
6	0	0	0.2175	θ_6

After this formulation, we can decouple into IK for the base and then IK for the 6DOF arm. For inverse kinematics for Arm, we used **Levemberg-Marquadt Numerical IK**.

2.1.2 Motion Planning

Once we receive the goal configuration in Configuration Space, we utilize Sampling-Based planners to chart a path in continuous space for both the mobile base and the mobile manipulator. The process commences by collecting the current joint angles and mobile base odometry. The planners then generate a path to the goal configuration by employing the Combined IK formulation for the specified goal point in task space.

Four prominent Sampling-Based planners were utilized, namely RRT, RRT*, Bi-RRT, and Bi-RRT*. Notably, RRT* and Bi-RRT* provide close to optimal solutions, with BiRRT* offering a higher density of node samples due to the growth of two trees. Although RRT and Bi-RRT also yielded solutions, they were not as optimal (shortest path). The incorporation of four planners offers users the flexibility to select the most suitable planner for specific scenarios.

2.1.3 Collision Avoidance

The true test of Motion Planners lies in their ability to successfully discover a secure, collision-free path for the robot. Typically, the conventional method involves utilizing sensors such as cameras to gauge the size and location of objects in the environment, enabling planners to navigate while considering obstacle positions. However, in this project, the objective was to eliminate the dependence on external sensors and instead rely solely on the collision feedback provided by the physics engine.

Problems and Issues: As we opted to rely on the physics engine for obtaining contact details between the manipulator and objects, we encountered certain challenges.

- The Gazebo Classic Physics Environment does not support contact feedback information, meaning it lacks the necessary tools to pre-check for potential collisions. Instead, it only provides visual feedback, which proved to be inadequate for our purposes.
- Determining the location of obstacles in the environment without the use of external sensors was a major concern. This issue posed a significant obstacle in our path planning efforts.

Solutions and Assumptions: We surfed through some good ideas to tackle the aforementioned issues and also that can serve our purpose and requirements

- To overcome the limitations of the Gazebo Classic Physics Environment, we used the Bullet Physics Engine as an intermediary to obtain collision feedback between the manipulator and obstacles. By utilizing the PyBullet API, we were able to check for collisions of sampled configurations before incorporating them into the final path plan.
- To ascertain the location and dimensions of the object, we relied on the model states provided by Gazebo, which are accessible through the ***/gazebo/model_states*** topic.

2.1.4 Control & Localization

This project encompasses the extensive use of Sampling-Based Planners, an Extended Inverse Kinematics Planner, and Collision Avoidance utilizing a completely separate Physics Engine. To accommodate the Control and Localization requirements of the Youbot, we have utilised some prominent settings within Gazebo and ROS.

Control:

- To streamline the control process, we integrated the ROS Control package, specifically employing position controllers for each manipulator actuator. This setup allows the Planner to simply supply the desired joint angles to these topics, enabling us to monitor the motion within Gazebo effectively.
- In the case of the Mobile Base, we utilized the ***cmd_vel*** topic along with a finely tuned proportional controller. This controller converts the position data into linear and angular velocity, which are calculated by the Combined Inverse Kinematics planners.

Localization: The localization accuracy of the Mobile Base was considered to be accurate. We relied on the ***/odom*** topic to gather information regarding the Mobile Base's position. Similarly, the current location of the Manipulator was obtained from the ***/youbot/joint_states*** topic.

2.2 Implementation

The flow chart shows how the flow of our algorithm occurs.

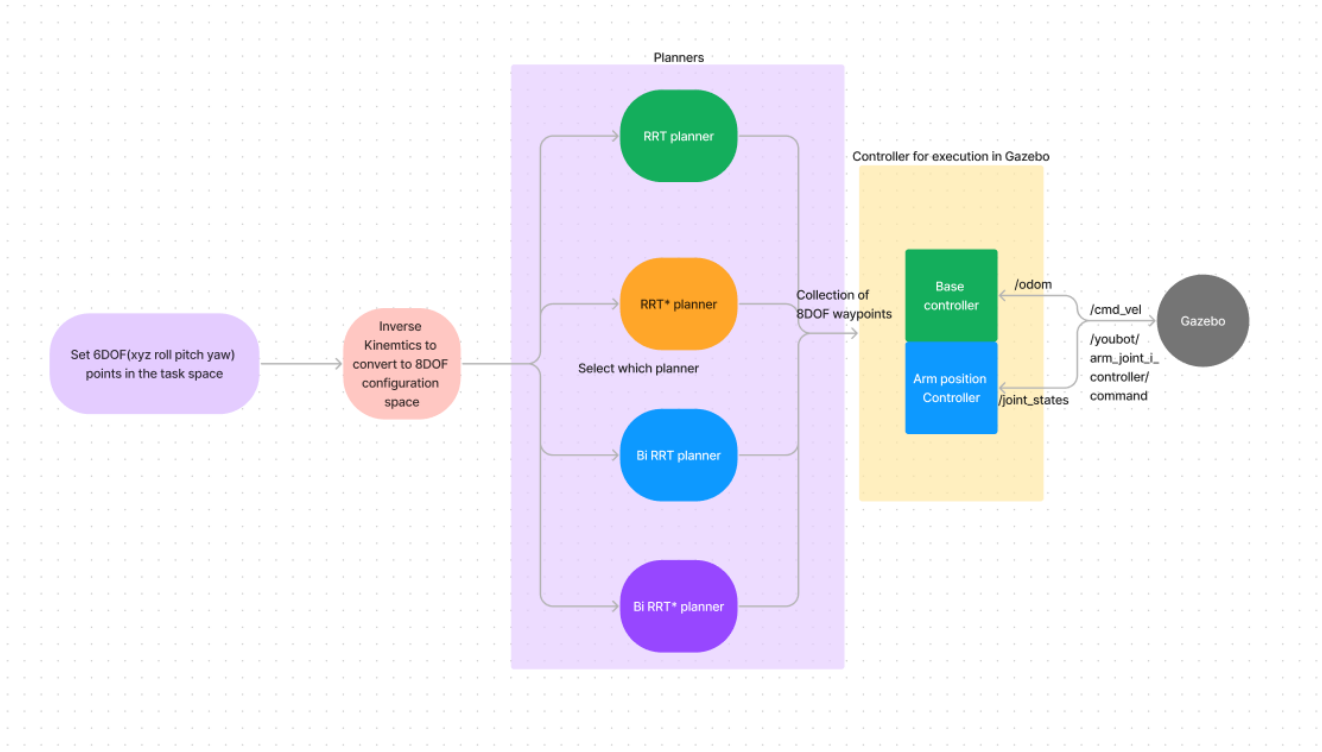


Figure 3: Flowchart of the proposed solution

We used Gazebo to simulate our robot. For performing collision checking for planning, we used pybullet. For forward and inverse kinematics, we used the robotics toolbox for Python to do the same.

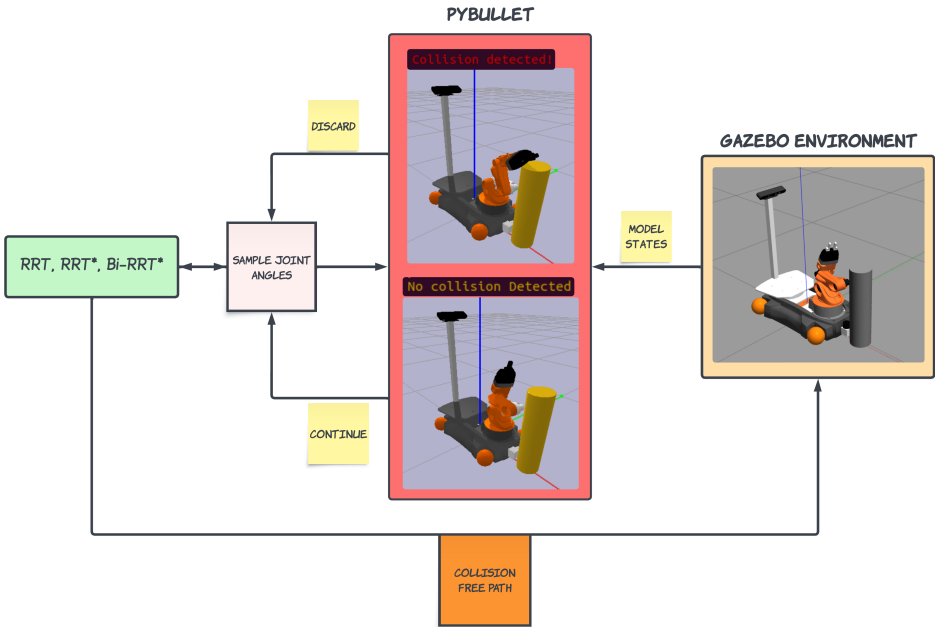


Figure 4: Collision Avoidance Flowchart

2.3 Challenges

Since we were using an Mobile Manipulator System, planning in Task Space is quite hard. We had numerous Challenges on the way, which had some great solutions but with some shortcomings

1. **Challenge 1:** Our primary challenge involved devising the Inverse Kinematics for the Mobile Manipulator system. Given its redundant nature, determining the inverse kinematics posed a significant challenge. We referred numerous research papers to explore potential solutions to this issue. Many studies employed the augmented Jacobian approach to address similar problems. However, when we applied this method to the 7-degree-of-freedom (DOF) system, it encountered failures in various scenarios.

   Solution : We devised a novel concept of augmenting the joints with a pseudo joint, which proved successful for us. This innovative approach set us apart, as it was not commonly found in existing research papers.

2. **Challenge 2:** Our subsequent hurdle revolved around the limited workspace of the Manipulator. The Youbot's 5-degree-of-freedom (DOF) Manipulator possesses a compact workspace attributed to its link lengths and joint constraints. At times, even the most intuitive plans to reach the goal point were disrupted by either the constraints of joint limits or the risk of self-collision.

   Solution : The fundamental strategy we adopted involved utilising our Combined IK Formulation to address configurations requiring movement of the Mobile Base. The output from the IK Solver yielded a floating-point sequence of length 8. The initial six values represented the Manipulator's Joints, including the Pseudo Joint, while the remaining two denoted the coordinates to be reached by the Mobile Base.

3. **Challenge 3:** Despite being a user-friendly simulator, Gazebo presented us with additional challenges. Particularly, the collision checking functionality lacked detailed feedback on mesh collisions, treating them as basic collision issues. Instead of pinpointing where two meshes collide, Gazebo visually displaced the obstacles and only issued a warning in the terminal.

   Solution : To overcome the limitations of Gazebo's collision checking, we utilised the Bullet Physics Engine. This allowed us to obtain mesh-mesh collision feedback before the obstacles were physically displaced, providing us with more detailed information about potential collisions.

3 Results

The image below illustrates the joint angles achieved by each controller after receiving input from the Planner. Five distinct goal positions were provided, and the plot displays the outputs of all five controllers, each corresponding to a joint of the Youbot's Manipulator. Our planner successfully generated a smooth trajectory towards the goals, evident from the linear progression of each line. However, one goal position required reaching through safe intermediate steps, as indicated by the curved segments in the plot's third section.

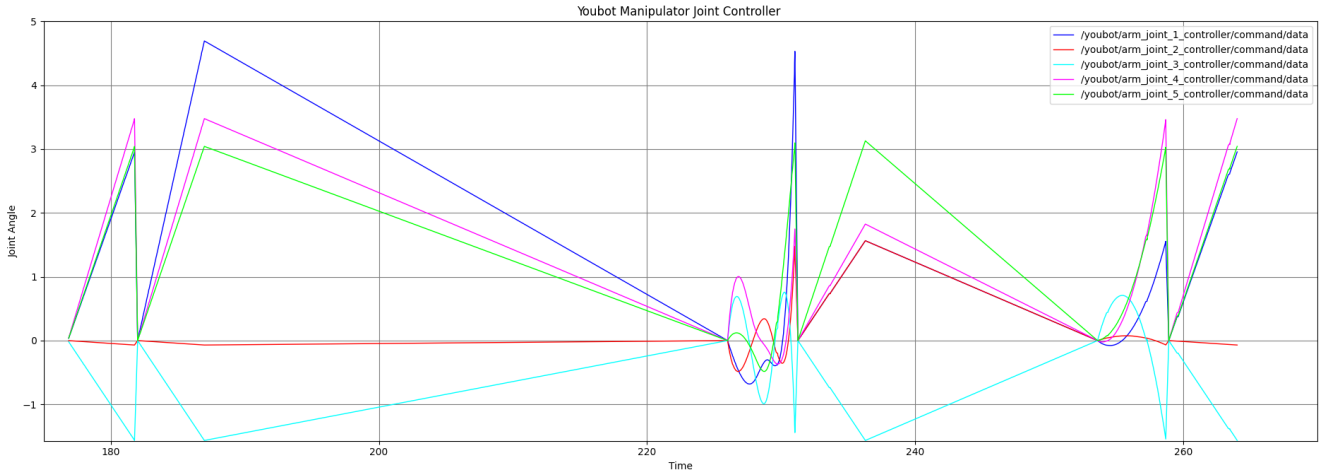
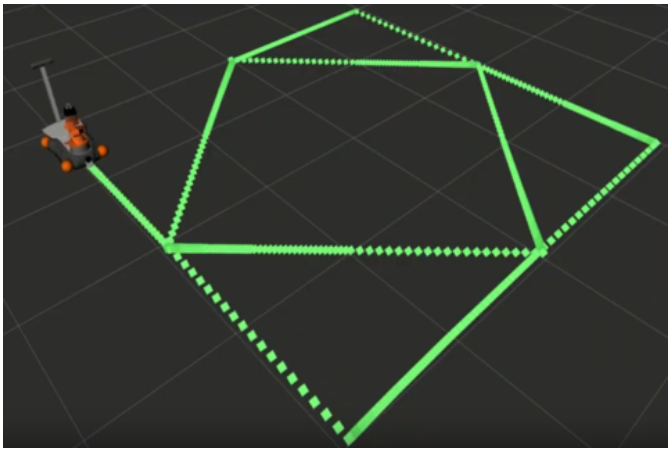


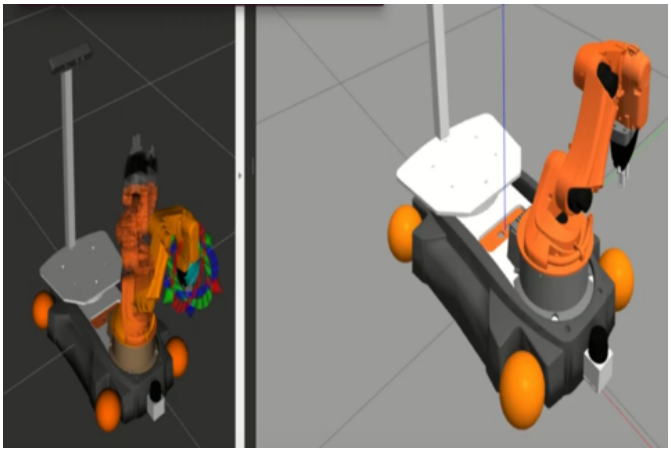
Figure 5: Joint Controller Trajectory Plot



This image depicts the trajectory traced by the Mobile Base using the RRT* Planner.

The base follows an inner square of dimensions 2.8m x 2.8m and an outer square of 4m x 4m. The green marker represents the update of the base odometry in the x-y plane.We increase the bias for this since there were no obstacles in the environment, this caused the goal to be sample very quicky and very few waypoints in planned path.The velocity controllers interpolates linear path between 2 waypoints , we get a path vey close to the optimum path. But this may not suit the actual scenario we are in, so we decreased the bias in the future planners to take into consideration the environment with obstacles.

Figure 6: Trajectory Traced by RRT* Planner



This image showcases the comparison between our RRT* Planner and the one employed by the MoveIt Planning Framework.

The generated trajectory is nearly identical and closely resembles the one found by the MoveIt Planner.

Figure 7: Comparison between MoveIt and our RRT* Planner

The screenshots below illustrate the comparison between Bi-Directional RRT and Bi-Directional RRT* as they trace the triangle path. Bi-RRT* follows a smooth trajectory, while the former traces a disrupted or curved path.

Figure 8: Bi-Directional RRT

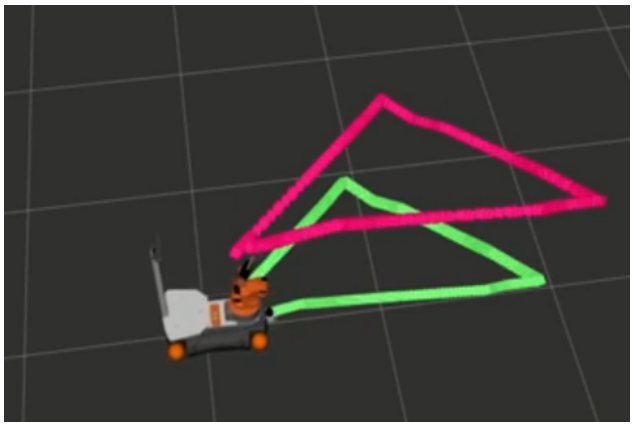
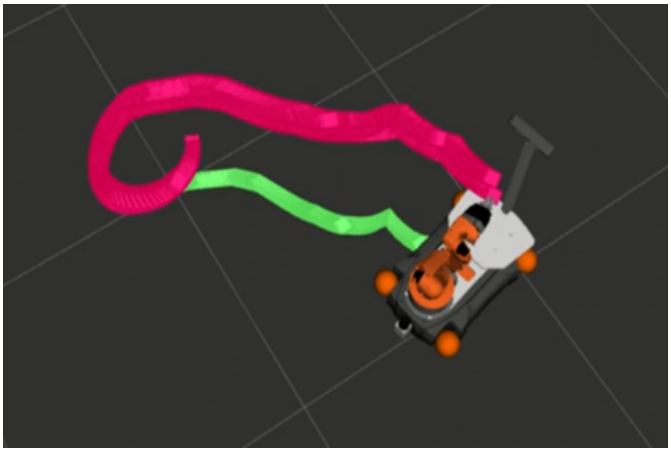


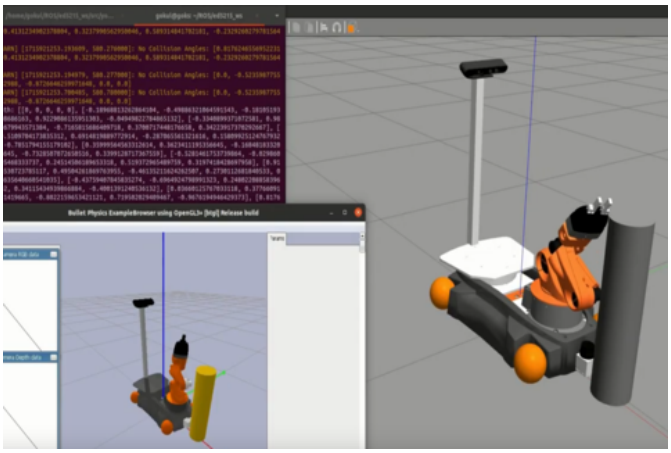
Figure 9: Bi-Directional RRT*



This image demonstrates the path followed when employing the Extended Inverse Kinematics Solver. Each point along the trajectory is transformed from Task Space to Joint Space and Base Positions, then provided to the corresponding controllers for motion control.

The solution comprises a floating sequence of 8 elements. We exclude the pseudo joint, with the remaining 7 utilized for planning purposes.

Figure 10: Combined Inverse Kinematics 8-DOF System



This image illustrates the complete pipeline, starting from computing the Inverse Kinematics Solution, feeding it into the Motion Planner, and utilizing the Bullet Physics Engine to check for collision-free samples.

The final collision-free path is then sent to the respective Controllers to visualize the trajectory in the Gazebo Environment.

Figure 11: Collision Avoidance using Bullet Physics Engine

The below image shows the full execution of the pipeline.The arm has to avoid collision with the obstacle on its plate and the base should avoid collision with the cuboid. The final end point of the end effector is [1,1,0.2].We can see that the end effector reaches the position by avoiding the obstacles which shows that our pipeline could handle static obstacle environment too.

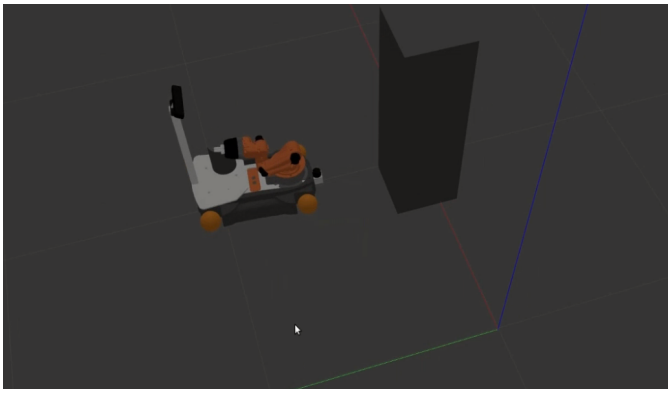


Figure 12: Entire pipleline demonstration

The image below showcases the joint angles attained by each controller after receiving a collision-free path from Pybullet. The visualization depicts the results of all five controllers, demonstrating a smooth transition in joint angles towards the target position without encountering any disruptions.

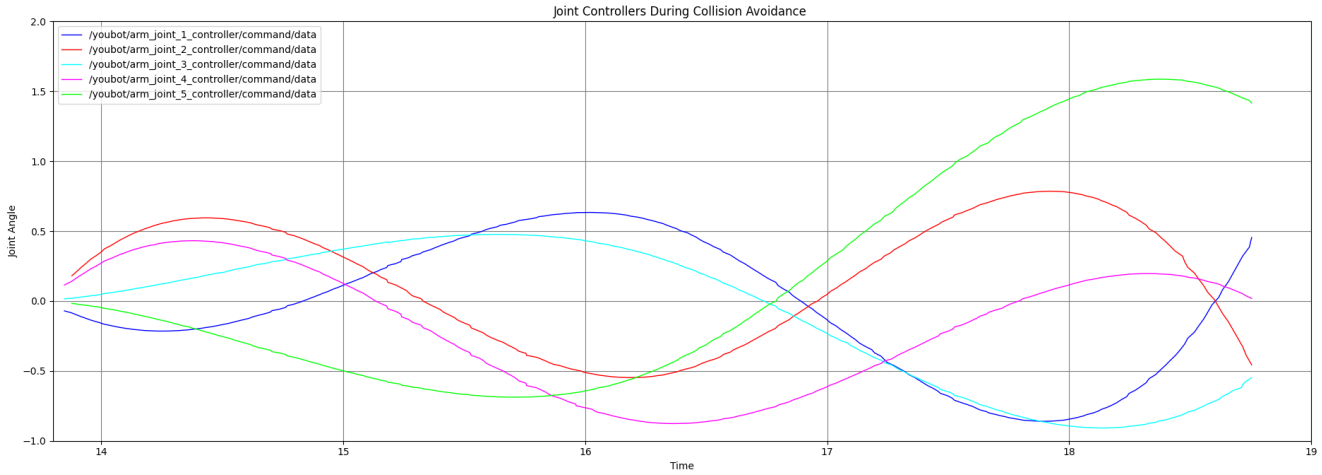


Figure 13: Joint Controller Trajectory Plot during Collision Avoidance

4 Discussion

One of the biggest advantages of our work is that it works even when the static obstacles are present in the environment whereas the traditional control approach used in solving the problem doesnt take into account the obstacles. Another great strenght of our work is the inverse kinematics. Previous work done for the inverse kinematics in various research papers uses a augmented jacobian approach. We tried to use such an approach but singularities and since the dof is 8 , it seemed challenging. To counter this we came up with a simple but powerful approach to solve this by decoupling the inverse kinematics to base plus pseudo joint manipulator. Pseudo manipulator is basically adding another joint at the base footprint and assuming the manipulator to be a 6dof arm instead of 5dof. This gives us more solutions to the inverse kinematics. This seemed to be very powerful and helps us to find the inverse kinematics for each of the trajectory waypoint and helped us to plan.

One of the weaknesses of our approach is planning time. Since we are planning in a 8dof space it takes more time to plan.Further more since we are using a sampling based algorithm, it shows great deviation from the optimum path(the path traced out by a human if he does the same task). So there is unecessary extra waypoints introduced which within 2 given waypoints which might not suit the task.The collision avoidance part works for simpler primitive shapes but when it becomes very irregular, it becomes challenging to find a path and plan.Using a physics engine like Pybullet helps in this but is computationally expensive.Currently the controllers used to control the arm and base works serially to help to visualize the path generated but ideally should be parallel.Our work does not involve dynamic obstacle which is very much required for real case usage scenarios and our pipleline would fail in such scenarios.

5 Contributions

- **Keerthi Vasan M:** Velocity controller for the base, BiRRT, BiRRT* implementation, Inverse Kinematics for the system,Combining the different modules and final execution(50%).
- **Gokul M K:** Simulation setup, Position Controllers for the arm, RRT, RRT* implementation, Collision checking implementation for the planners(50%).

References

[1] S. M. LaValle, *Planning Algorithms. Cambridge Univ. Press,*(2006), New York, NY, USA:

[2] LaValle, S. M., & Kuffner Jr, J. J. 2000, *Rapidly-exploring random trees: Progress and prospects.*

[3] H. Simas, A. Dias and D. Martins. (2012) *Extended Jacobian for redundant robots obtained from the kinematics constraints*. In G. G. Editor, H. H. Editor, & I. I. Editor (Eds.), *ABCM Symposium Series in Mechatronics - Vol. 5* (Section VII - Robotics Page 1005), https://abcm.org.br/symposium-series/SSM_Vol5/Section_VII_Robotics/04082.pdf

[4] P. I. Corke, *A robotics toolbox for MATLAB*, *in IEEE Robotics & Automation Magazine, vol. 3, no. 1, pp. 24-32, March 1996*, https://doi.org/10.1109/100.486658

[5] Qi, J., Yuan, Q., Wang, C. et al. (2023). *Path planning and collision avoidance based on the RRT*FN framework for a robotic manipulator in various scenarios*, *Complex Intell. Syst. 9, 7475–7494 (2023)*, https://doi.org/10.1007/s40747-023-01131-2

[6] Quang-Nam Nguyen and Quang-Cuong Pham (2024). *Planning Optimal Trajectories for Mobile Manipulators under End-effector Trajectory Continuity Constraint*, https://arxiv.org/abs/2309.12251

[7] Sustarevas, Julius & Kanoulas, Dimitrios & Julier, Simon. (2021). *Task-Consistent Path Planning for Mobile 3D Printing*. https://10.1109/IROS51168.2021.9635916

[8] G. Oriolo and C. Mongillo (2005) *Motion Planning for Mobile Manipulators along Given End-effector Paths. Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain*. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1570432