Trajectory Tracking for Multiactuated Bots
ME 494: Bachelors Thesis Project 1

Manan Mehta(22b2129)

Contents

Acknowledgements

I would like to express my sincere gratitude to Prof. Anirban Guha and Prof. Arpita Sinha for their guidance throughout the course of this thesis. Prof. Arpita Sinha provided clear direction and valuable feedback that helped refine both the technical content and its presentation. Prof. Anirban Guha offered crucial insights, constructive discussions, and consistent support, all of which significantly shaped the development of this work.

I am also thankful to my peers, friends, and family for their encouragement and understanding during the completion of this project.

Abstract

This thesis investigates the problem of coordinated trajectory tracking for a multi-actuated robotic system consisting of mobile omnidirectional bases equipped with planar revolute-prismatic-revolute (R-P-R) manipulators. The objective is to enable a team of robots to cooperatively manipulate and transport a rigid object along a prescribed trajectory while maintaining continuous contact, enforcing geometric constraints, and avoiding inter-robot collisions.

A kinematic model for the bot-arm system is developed, followed by a formal description of object-bot interaction under rigid-body constraints for both two and multiple end-effector configurations. Two Model Predictive Control (MPC) formulations are proposed. The first formulation employs angle-based constraints but encounters numerical instability due to discontinuities introduced by the atan2 function. The second formulation re-parameterizes object orientation using linear constraints on $\cos(\theta)$ and $\sin(\theta)$, resulting in improved numerical behavior. Simulation results presented here demonstrate successful trajectory tracking and highlight the effect of the control effort penalty R on smoothness and stability of robot motion.

The work establishes a foundation for extending cooperative manipulation to higher-dimensional tasks and larger robot teams. It also prepares the groundwork for transitioning from a centralized MPC architecture to a decentralized reinforcement-learning-based controller, which is proposed as future work.

Defining the Problem

To define the problem, the report first define the constituent moving parts of the entire system and define how they affect each other.

3.1 Bot

In this pdf, 'bot' has loosely been used to address the base of the mobile robot + the arm effector mounted on top of it. The bot is omnidirectional in nature and can move in any direction in the 2d plane. The arm effector mounted on top of it can also move in a revolute-prismatic-revolute manner to position the end-effector at any point in the 2d plane.
Inputs: x, y, max speed (v)
Other possible expected inputs for the bot will include: bot_base x,y and the height at which the arm effector has to remain at. A detailed design (including the urdf) will bring in more paramaters for the inverse kinematic positioning of the end effector.

3.2 Object

The 'object' refers to the particular item that has to be transported by the bots from one location in space to the other. The definition of the object requires particular geometries assigned to it, as that can be attributed to making the bot move in particular orientations in the 2d plane. Further, an analysis into 3d movements subsequently will potentially lead to differences in z-orientations for the end effectors.

3.3 Destination and/or Trajectories

The problem at hand assumes a trajectory with some randomly assigned velocities for the center of mass of the object to follow. The different bots will have different speeds so that the object can have necessary speeds as well as angular velocities.

3.4 Object-Bot Interaction

The bots push the object and can move omni-directionally in the x-y plane. if both bots move in the same direction, the object translates, if one moves, other doesn't it's rotation. Thus, this is essentially a rigid body kinematics problem. Thus, what is needed is probably to know how and where the bots pick the obj ect up.

Apart from this, the distance between the gripper of the bot and the object's COM: $\sqrt{(x_{\text{bot, gripper}} - x_{\text{COM of object}})^2 + (y_{\text{bot, gripper}} - y_{\text{COM of object}})^2}$ as well as the angle formed between any three points on the object is constant. i.e. r and θ of bot's gripper relative to object is constant for the current problem.

3.4.1 2 Arm Effectors - Object interaction

To describe the kinematics of the object (subsequently referred to as box or object), it is quite evident that the entire object + end-effectors be treated as rigid bodies. Since the bots always maintain contact with the object at all times, the speed at which the bots move is equivalent to the local points on the object.

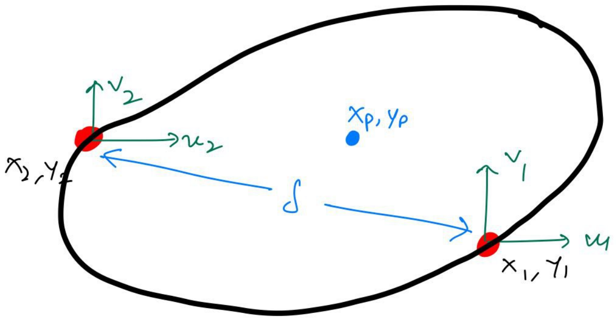


Figure 3.1: Schematic of the object with the red dots being the end-effectors of the bots

To start off with, we define the coordinates of the Center of Mass (COM) of the body as well as the positions of the bots with respect to an arbitrarily placed fixed frame of reference. Let the COM of the body be $\{x_p, y_p\}$ and the centers for the grippers be $\{x_{G,1}, y_{G,1}\}$ and $\{x_{G,2}, y_{G,2}\}$ respectively. The control inputs for such systems lie with the bot's grippers and manipulators. Thus, we can effectively control the velocities and angular velocity(ω) for the end-effector of 1 bot-effector system and thus calculate velocities (u_p, v_p) for the COM accordingly.

$$\begin{bmatrix} u_{G,1}\hat{\imath} + v_{G,1}\hat{\jmath} \\ u_{G,2}\hat{\imath} + v_{G,2}\hat{\jmath} \end{bmatrix} = (u_p\hat{\imath} + v_p\hat{\jmath}) + \vec{\omega} \times \begin{bmatrix} \vec{r_1} - \vec{r_p} \\ \vec{r_2} - \vec{r_p} \end{bmatrix} \tag{3.1}$$

Solving u_P, v_P using arm effectors 1 and 2, we get:

$$u_p = u_{G,1} + \omega(y_{G,1} - y_p) = u_{G,2} + \omega(y_{G,2} - y_p) \tag{3.2}$$

$$v_p = v_{G,1} - \omega(x_{G,1} - x_p) = v_{G,2} - \omega(x_{G,2} - x_p) \tag{3.3}$$

Thus, u_1 and u_2 are related to each other by ω and so are v_1 and v_2

3.4.2 Multiple Arm Effectors - Object interaction

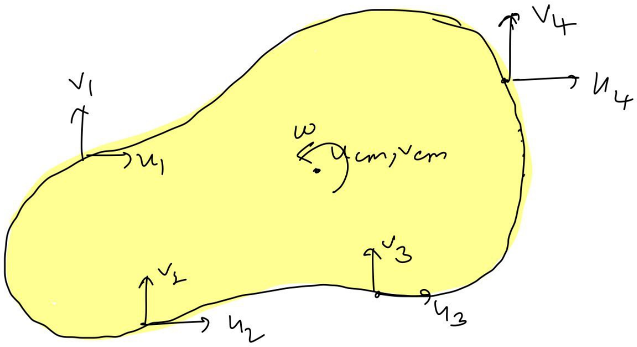


Figure 3.2: Object with multiple end Effectors on its surface

$$\begin{bmatrix} u_1\hat{i}+v_1\hat{j} \\ u_2\hat{i}+v_2\hat{j} \\ \dots \\ u_N\hat{i}+v_N\hat{j} \end{bmatrix} = (u_p\hat{i}+v_p\hat{j}) + \vec{\omega} \times \begin{bmatrix} \vec{r_1}-\vec{r_p} \\ \vec{r_2}-\vec{r_p} \\ \dots \\ \vec{r_N}-\vec{r_p} \end{bmatrix} \tag{3.4}$$

This is essentially multiple equations but limited variables and hence for a solution set to exist, the solution set will lie in the Null space of the 2-arm effector solution set. In a more physical sense, since all of these arms lie on the rigid body of the object, it is constrained to move in certain ways. i.e. based on any 1 effector, and known locations of the other arm effectors in respect to any two arms or the center of mass of the object, the motion of the other arms will be pre-determined essentially. If this is not followed, there will be internal stresses developed within the object and can lead to deformations (local and/or global elongations and compressions)

3.5 Bot-Arm Interaction

Although the actual sytem for the bot is in 3 Dimensions (Revolute - Revolute - Revolute joints), it can be translated to 2d as a Revolute - Prismatic - Revolute joint. Now, we shall define what are the variables that we can control and further what variables describe the state.

3.5.1 Setting up of the problem:

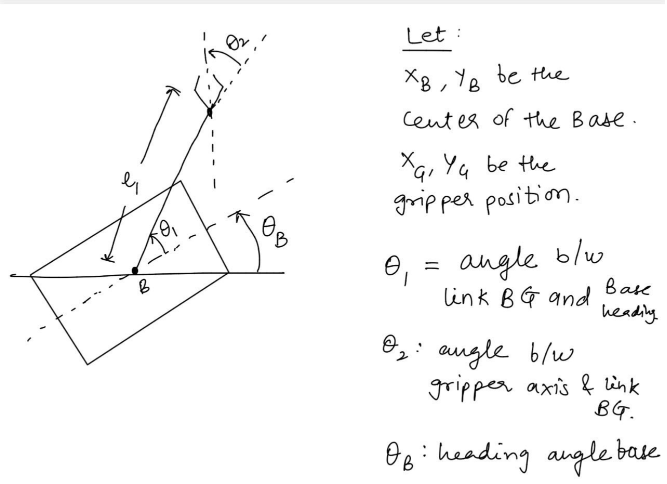
Define the following for Bot A's state:

- θ_B is the heading angle for the base of the robot
- x_B is the x coordinate for the centroid of the base of the robot
- y_B is the y coordinate for the centroid of the base of the robot
- θ_1 is the angle that the arm makes with the bot's base's heading
- l_1 is the length of the arm of the bot (variable since 2d projection)

- θ_2 is the angle that the gripper makes with the arm

The control inputs for the bot A are:

- u_B is the x velocity for the centroid of the base of the robot
- v_B is the y velocity for the centroid of the base of the robot
- ω_B is the angular velocity of the base of the robot
- ω_1 is the angular velocity of the arm wrt base
- $\dot{l_1}$ is the kinematic rate of change for the length of the arm.
- ω_2 is the angular velocity of the gripper wrt arm



Let:

x_B, y_B be the Center of the Base.

x_G, y_G be the gripper position.

θ_1 = angle b/w link BG and Base heading

θ_2: angle b/w gripper axis & link BG.

θ_B: heading angle base

Figure 3.3: A schematic of the bot-arm system

There also exists the following constraints on the system:

- $l_0 \leq l_1 \leq l_{max}$
- The orientation of the gripper wrt to the object must stay constant. Mathematically: $\omega_{\text{object}} = \omega_{\text{gripper 1}} = \omega_{\text{gripper 2}}$

3.5.2 Forward Kinematics

$$x_G = x_B + l_1 \cos(\theta_B + \theta_1)$$
$$y_G = y_B + l_1 \sin(\theta_B + \theta_1)$$
$$\theta_G = \theta_B + \theta_1 + \theta_2$$

3.5.3 Inverse Kinematics

Currently, the knowns are the positions and orientations of the gripper in respect to a global coordinate system. Further, we wish to incorporate the velocities and ω of the gripper to plan the path for the object.

$$\Delta x = x_G - x_B \quad ; \quad \Delta y = y_G - y_B$$

$$\theta_1 = \arctan 2(\Delta y, \Delta x) - \theta_B(\text{Heading Angle})$$

$$l_1 = \sqrt{(\Delta x)^2 + (\Delta y)^2}$$

On resolving thse and calculating their derivatives, we can state the following:

$$\begin{bmatrix} \dot{x}_G \\ \dot{y}_G \\ \dot{\theta}_G \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cos(\theta_1 + \theta_B) & -l\sin(\theta_1 + \theta_B) & -l\sin(\theta_1 + \theta_B) & 0 \\ 0 & 1 & \sin(\theta_1 + \theta_B) & l\cos(\theta_1 + \theta_B) & l\cos(\theta_1 + \theta_B) & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_{B,1} \\ \dot{y}_{B,1} \\ \dot{l}_1 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_{B,1} \end{bmatrix} \tag{3.5}$$

MPC Formulation

For the duration of the project, two approaches were tried. The following is the two different forms of formulations for the Model Predictive Control based controller for the bots:

4.1 Formulation 1

4.1.1 Initial Conditions and other Knowns

We know the state at t=0: thus, we know X = [x_B^1, y_B^1, q_1^1, q_2^1,q_3^1, x_B^2, y_B^2, q_1^2, q_2^2,q_3^2] and the location of the center of mass of the object (x_c, y_c)

4.1.2 Formulation

Objective Function:

$$\min J(x) = J_1(x) + J_2(x) + J_3(x) \tag{4.1}$$

Where, $J_1(x)$ is the trajectory tracking cost.

$$J_1(x) = \sum_{k=i}^{i+N-1} \begin{bmatrix} x_k - X_{\text{ref,k}} \\ y_k - Y_{\text{ref,k}} \end{bmatrix}^T \begin{bmatrix} P_x & 0 \\ 0 & P_y \end{bmatrix} \begin{bmatrix} x_k - X_{\text{ref,k}} \\ y_k - Y_{\text{ref,k}} \end{bmatrix}$$

J_2 is the effort minimisation cost.

$$J_2(x) = \sum_{j=1}^{10} \sum_{k=i}^{i+N-1} u_{j,k}^T R_{ca} u_{j,k}$$

J_3 is the inter robot collision avoidance cost.

$$J_3(x) = \sum_{k=0}^{N-1} \begin{bmatrix} x_{\text{Bot1 base, k}} - x_{\text{Bot2 base, k}} \\ y_{\text{Bot1 base, k}} - y_{\text{Bot2 base, k}} \end{bmatrix}^T * Q * \begin{bmatrix} x_{\text{Bot1 base, k}} - x_{\text{Bot2 base, k}} \\ y_{\text{Bot1 base, k}} - y_{\text{Bot2 base, k}} \end{bmatrix}$$

Constraints and Equations

Control Constraint 1
Based on the Initial Conditions, we define the following constants:

$$l_0 = \sqrt{(x_c - x_{g,1})^2 + (y_c - y_{g,1})^2}; \quad \theta_0 = \text{atan2}\left(\frac{y_{g,2} - y_{g,1}}{x_{g,2} - x_{g,1}}\right) - \text{atan2}\left(\frac{y_c - y_{g,1}}{x_c - x_{g,1}}\right)$$

Where, $(x_{g,i}, y_{g,i})$ represents the (x,y) coordinates of the gripper for Bot i and (x_c, y_c) represents the center of mass for the object. Using this, if the locations of grippers are known at any moment t, we can write the relation for coordinates of C (object's center of mass) as:

$$C_t \equiv \left\{ x_{g,1} + l_0 \cos\left(\text{atan2}\left(\frac{y_{g,2} - y_{g,1}}{x_{g,2} - x_{g,1}}\right) - \theta_0 \right), \quad y_{g,1} + l_0 \sin\left(\text{atan2}\left(\frac{y_{g,2} - y_{g,1}}{x_{g,2} - x_{g,1}}\right) - \theta_0 \right) \right\}\Bigg|_t \tag{4.2}$$

Control Constraint 2
Further define the system as:

$$X^i = \begin{bmatrix} x_B \\ y_B \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}^i \; ; \; U^i = \begin{bmatrix} u_{B,x} \\ u_{B,y} \\ \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix}^i \implies \dot{X^i} = U^i \tag{4.3}$$

Control Constraint 3

Distance between bots must be greater than some threshold.

$$d(B_1, B_2) >= \text{d} \tag{4.4}$$

(Also, note, it would be beneficial to have $d(B_1, B_2)$ to be as low as possible)

4.1.3 Results

The above formulation doesn't lead to any non trivial solutions as it runs into a lot of numerical issues due to the use of atan2 function. Thus, the work on this formulation had to be discontinued.

4.2 Formulation 2

4.2.1 Initial Conditions and other Knowns

We know the state at t=0: thus, we know X = [x_B^1, y_B^1, q_1^1, q_2^1,q_3^1, x_B^2, y_B^2, q_1^2, q_2^2,q_3^2] and the location of the center of mass of the object (x_c, y_c)

4.2.2 Cost function

Define:

$$\min J(x) = J_1(x) + J_2(x) + J_3(x) \tag{4.5}$$

Where, $J_1(x)$ is the trajectory tracking cost.

$$J_1(x) = \sum_{k=i}^{i+N-1} \begin{bmatrix} x_k - X_{\text{ref,k}} \\ y_k - Y_{\text{ref,k}} \end{bmatrix}^T \begin{bmatrix} P_x & 0 \\ 0 & P_y \end{bmatrix} \begin{bmatrix} x_k - X_{\text{ref,k}} \\ y_k - Y_{\text{ref,k}} \end{bmatrix}$$

J_2 is the effort minimisation cost.

$$J_2(x) = \sum_{j=1}^{10} \sum_{k=i}^{i+N-1} u_{j,k}^T R_{ca} u_{j,k}$$

J_3 is the inter robot collision avoidance cost.

$$J_3(x) = \sum_{k=0}^{N-1} \begin{bmatrix} x_{\text{Bot1 base, k}} - x_{\text{Bot2 base, k}} \\ y_{\text{Bot1 base, k}} - y_{\text{Bot2 base, k}} \end{bmatrix}^T * Q * \begin{bmatrix} x_{\text{Bot1 base, k}} - x_{\text{Bot2 base, k}} \\ y_{\text{Bot1 base, k}} - y_{\text{Bot2 base, k}} \end{bmatrix}$$

and N is the horizon.

4.2.3 Formulation

Control Constraint 1

Instead of the previous formulation, we now move on to solve the angles and translations through the solving for the $\cos(\theta), \sin(\theta), t_x, t_y$. Given initial points O, A, B, and final points A' and B';
$A'_{k+1} = R_k.A_k + t_k$;
$B'_{k+1} = R_k.B_k + t_k$;

solving for

$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}; \quad t = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

which yields the overall equations:

$$\begin{bmatrix} A_x & A_y & 1 & 0 \\ A_y & -A_x & 0 & 1 \\ B_x & B_y & 1 & 0 \\ B_y & -B_x & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta \\ \sin\theta \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} A'_x \\ A'_y \\ B'_x \\ B'_y \end{bmatrix} \tag{4.6}$$

Solving for $\cos\theta, \sin\theta, t_x, t_y$ and using that to solve for O:
O' = R . O + t

Control Constraint 2

Now, we also have system as:

$$X^i = \begin{bmatrix} x_B \\ y_B \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}^i ; \ U^i = \begin{bmatrix} u_{B,x} \\ u_{B,y} \\ \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix}^i \implies \dot{X^i} = U^i \tag{4.7}$$

Further, compile multiple Bots into a single X vector:

$$X = \begin{bmatrix} \text{Bot } 1_x \\ \text{Bot } 2_x \\ \vdots \\ \text{Bot N}_x \end{bmatrix}; U = \begin{bmatrix} \text{Bot } 1_q \\ \text{Bot } 2_q \\ \vdots \\ \text{Bot N}_q \end{bmatrix} \implies \dot{X} = U$$

Control Constraint 3

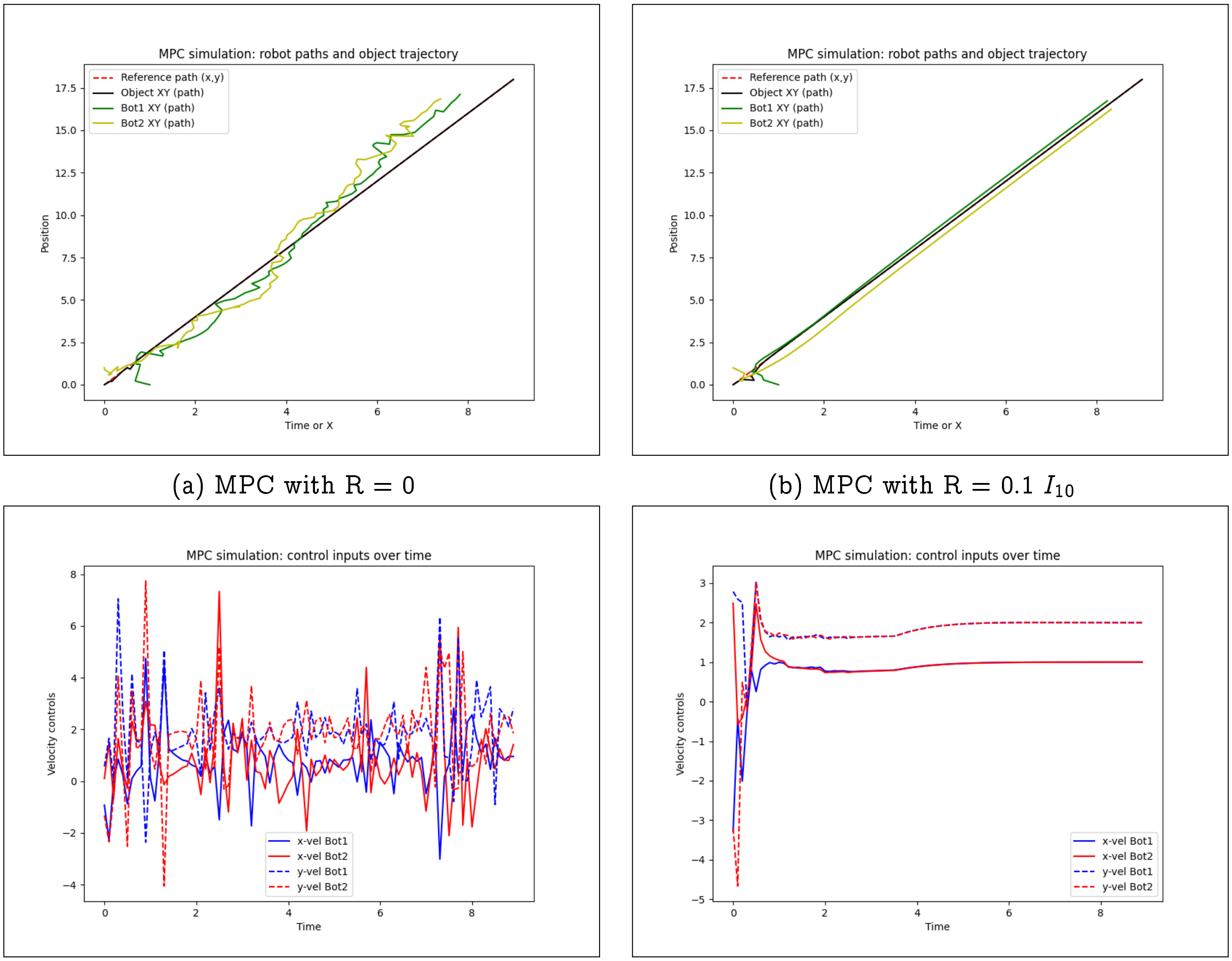
Distance between bots must be greater than some threshold.

$$d(B_1, B_2) >= \mathrm{d} \tag{4.8}$$

(Also, note, it would be beneficial to have $d(B_1, B_2)$ to be as low as possible.)

4.2.4 Results

For: P $=$ 20 I_{10}, Q $=$ 10 I_{10} (10 is the size of nu)



(a) MPC with R $=$ 0

(b) MPC with R $=$ 0.1 I_{10}

(c) Controls with R=0

(d) Controls with R $=$ 0.1 I_{10}

Figure 4.1: Results for trajectory tracking with varying R

Conclusion

This thesis presented the modelling and control of a multi-actuated cooperative manipulation system consisting of omnidirectional mobile robots equipped with planar R–P–R manipulators. A complete kinematic description of the bot–arm system and its interaction with a rigid object was developed in the 2D plane. Two Model Predictive Control (MPC) formulations were constructed to enable coordinated trajectory tracking of the object while respecting contact constraints and preventing inter-robot collisions.

The first formulation, which directly incorporated object orientation through the atan2 function, was found to be numerically unstable due to inherent discontinuities in angular representation. This motivated the development of a second formulation that instead employs a linear parametrization of orientation using $\cos\theta$ and $\sin\theta$, thereby eliminating discontinuities and significantly improving solver behaviour. Simulation results demonstrate that this formulation enables stable and accurate cooperative manipulation and that the choice of the control-effort penalty R has a substantial effect on smoothness of control inputs and tracking stability.

The results obtained in this work provide a strong foundation for extending cooperative manipulation to higher-dimensional tasks, larger robot teams, and more cluttered operational environments. Moreover, the formulation developed here lays the groundwork for transitioning from a centralized MPC-based architecture to decentralized reinforcement-learning-based control strategies envisioned for the next stage of the project.

Future Work

The future work planned for BTP-2 builds upon the modelling, MPC formulation, and simulation framework developed in this thesis. Several important extensions and improvements are required to further advance the system towards robust multi-robot cooperative manipulation.

6.1 3D Formulation

To extend the system from 2D planar manipulation to full 3D cooperative manipulation, it is essential to model rigid-body transformations in SE(3). A complete description of 3D object motion requires tracking the coordinates of at least three non-collinear points on the object. The object's orientation must be represented using the composition of rotations about the x, y, and z axes, together with a 3D translation.

$$
\begin{aligned}
A'_{k+1} &= R_{z\,k}\, R_{y_k}\, R_{x\,k}\, A_k + T_k \\
B'_{k+1} &= R_{z\,k}\, R_{y_k}\, R_{x\,k}\, B_k + T_k \\
C'_{k+1} &= R_{z\,k}\, R_{y_k}\, R_{x\,k}\, C_k + T_k
\end{aligned}
$$

In expanded matrix form, this corresponds to:

$$
\begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix}_{k+1} = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix}_k \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix}_k \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix}_k \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix}_k + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}_k
$$

Developing a complete 3D MPC framework requires incorporating these SE(3) transformations into the prediction model, enforcing multi-point contact constraints in 3D, and addressing the increased dimensionality and nonlinearities of the control problem. This extension represents a major next step toward general, real-world cooperative manipulation.

6.2 Issues in 2D

While the 2D MPC formulation developed in this thesis successfully demonstrates basic cooperative manipulation, several limitations remain that must be addressed in future work:

1. **Robot Path Disparity and Path Crossing:** There exists a disparity in how the MPC computes robot trajectories. Specifically, the two bots cross a single point at a certain instance during initialization, a behaviour that must be eliminated to ensure safe and realistic operation. This may require improved collision-avoidance constraints, better initialization strategies, or modifications to the MPC horizon and cost structure.

2. **Scaling to More Robots and Cluttered Environments:** As the system is extended to a larger number of robots, the MPC must ensure coordinated motion without collision in increasingly cluttered or constrained spaces. This requires scalable constraint handling, improved obstacle-avoidance formulations, and potentially hierarchical or decentralized control structures to manage computational complexity.

Additional considerations—including visualizing trajectories with start/end markers, incorporating collision checking between the bots and the object, and generating richer control plots—will further support debugging, interpretability, and real-world implementation.

Move To Reinforcement Learning

The current system and protocols operate using a centralized MPC controller. As part of the next stage of the project, the goal is to transition toward a decentralized reinforcement-learning-based system. Approaches such as Q-learning, multi-agent deep reinforcement learning, or decentralized policy-gradient methods may be suitable candidates for this transition.

To enable such a shift, it is crucial that the MPC-based physics system developed in this work remains robust, accurate, and fully validated. The MPC controller will serve as the ground truth for training or benchmarking RL controllers, and therefore must produce reliable behaviour across a wide range of trajectories and environments.

The long-term aim is to develop a decentralized decision-making framework in which each robot learns its own policy while maintaining coordinated object manipulation, robustness to perturbations, and scalability to larger robot teams and 3D motion.

List of Figures