Calvin Stahoviak

Candidate

Department of Computer Science

Department

This thesis is approved, and it is acceptable in quality and form for publication:

Approved by the Thesis Committee:

Melanie Moses , Chairperson

Matthew Fricke

Jonathon Slightam

Dynamic Admittance Parameterization for Non-Prehensile Multi-Robot Transport with Optimal Coordinated Planning

by

Calvin Stahoviak

B.S., Computer Science, University of New Mexico, 2022

THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science
Computer Science

The University of New Mexico
Albuquerque, New Mexico

December, 2025

Acknowledgments

I would like to thank Professor Melanie Moses, my committee chair, for taking me in as a graduate student and encouraging me at every step of the way.

I would also like to thank Research Associate Professor Matthew Fricke for guiding and shaping my knowledge of the research world.

And finally, I would like to thank Dr. Jonathon Slightam, who has always noticed and brought out my potential.

Dynamic Admittance Parameterization for Non-Prehensile Multi-Robot Transport with Optimal Coordinated Planning

by

Calvin Stahoviak

B.S., Computer Science, University of New Mexico, 2022

M.S., Computer Science, University of New Mexico, 2026

Abstract

The **Dyn**amic **A**dmittance Parameterization of Non-Prehensile **M**ulti-Robot Transport with **O**ptimal Coordinated Planning (DYNAMO) architecture offers a practical framework for cooperative payload transportation using two robots equipped with nonholonomic mobile bases and four-degree-of-freedom manipulators. Coordinated mobile manipulation is a difficult problem in robotics, and the non-prehensile case is even more challenging than its prehensile counterpart because the robot bases and the payload are dynamically coupled. DYNAMO adapts arm motion in response to interaction forces and generates coordinated base trajectories that account for this coupling. Robust payload transport is achieved through the combination of optimal planning and adaptive compliant control, which enables each robot to maintain appropriate contact forces. In hardware experiments with two Joint Integrated Admittance, Navigation, and Transport (JiANT) robots, DYNAMO yields longer and more reliable transport than static admittance- or position-based methods. These results provide a practical foundation for cooperative non-prehensile manipulation in domains such as logistics, construction, and hazardous material handling.

Contents

List of Figures

List of Tables

Glossary

Admittance Admittance is the regulation of motion in response to forces at the port of interaction (inverse of impedance).

Collective A collective is a type of distributed work in which agents are unaware of other agents yet do share a common goal.

Collaboration Collaboration is a type of distributed work in which agents are aware of other agents, have individual goals, but actively help each other achieve their respective objectives through complementary actions.

Cooperation Cooperation is a type of distributed work in which agents are aware of other agents and share a common goal.

Coordination Coordination is a type of distributed work in which agents are aware of other agents but do not share a common goal and their actions do not help one another.

Damping Damping is the extent to which an object resists velocity in response to an applied force.

Force A force is a push or pull upon an object resulting from the object's interaction with another object.

Impedance Impedance is the regulation of forces in response to motion at the port of interaction (inverse of admittance).

Inertia Inertia is the extent to which an object resists acceleration in response to an applied force.

Inverse kinematics Inverse kinematics is the mathematical process of calculating the variable joint parameters needed to place the end of a kinematic chain (such as a robot manipulator) in a given position and orientation.

Kinematics Kinematics is the branch of mechanics concerned with the motion of objects without reference to the forces which cause the motion.

Manipulation Manipulation refers to the the mechanical interaction of object(s).

Multi-Robot System A multi-robot system is a robotic system in which two or more robots work together in a collective, collaborative, cooperative, or coordinated manner.

Non-Prehensile Manipulation Non-prehensile manipulation is the act of manipulating an object through pushing.

Port of Interaction The port of interaction is the contact point at which a system interacts with its environment.

Prehensile Manipulation Prehensile manipulation is the act of grasping and manipulating an object through gripping, clasping or other holding action.

Robot Description A robot description is a model or configuration file that describes the pose, size, and kinematic properties of a robot's links and joints.

(Hookean) Stiffness Hookean stiffness is the constant of proportionality that relates linear deformation to applied force.

Chapter 1

Introduction

Multi-robot transportation enables greater capability and adaptability compared to single-robot systems. These added benefits stem from the cooperative interactions between robots. Formally, a multi-robot system (MRS) is a system in which two or more robots work together to complete a given task [1]. A given task can be completed as a collective, with each robot working towards their own individual goal. However, the greater benefit of a MRS emerges when each robot cooperates with their peers. Although more beneficial, a MRS with cooperative robot-robot interactions is significantly more complex than one without interactions. Among the various common multi-robot tasks such as mapping [2], foraging [3], and surveillance [4], cooperative transportation of objects presents particular challenges in dynamics and control. In most manipulation tasks, a gripper end-effector is utilized to firmly grasp and hold the object. This type of manipulation, called prehensile manipulation, is attractive because once a firm grasp is established, the manipulator can reliably control the motion of the object and reasonably trust that the object will remain held without continuously monitoring the interaction forces or state of the object. However, this requires that the object have readily available grasping points, or be small enough to clasp. Immediately, prehensile manipulation greatly limits the

types of objects that can be transported. The transportation of irregular objects or objects without grasp-points requires the regulation of especially complex forces at the port of interaction, the point at which the end-effector interacts with the payload. Non-prehensile manipulation is the act of manipulating an object through controlled pushing. This type of manipulation can be generally applied to all types of objects. Although challenging and unconventional, the benefits of a cooperative MRS capable of non-prehensile manipulation and coordinated transportation can impact many domains.

The applications of cooperative transport include warehouse logistics, automated construction, disaster response, human–robot interaction, hazardous material handling, extraterrestrial construction, and in-situ resource utilization. For example, in a warehouse environment, large packages or barrels need to be efficiently rearranged or loaded. During disaster response, irregular and often grasp-free rubble needs to be moved. When transporting hazardous materials, a variety of containers may need to be moved in a safe and stable manner to avoid damage or harm to the environment. Although each use case involves objects of greatly varying size, shape and material, non-prehensile cooperative manipulation presents itself as a solution to many variations of the transportation task while maintaining stable and safe forces on the object.

A major challenge in cooperative transport is achieving success with minimal information [5]. Maintaining appropriate pressure and consistent contact with the payload is critical for non-prehensile manipulation. Even with accurate global pose estimates, small calibration errors, delays, and payload mechanics introduce uncertainty.

In the DYNAMO architecture, two mobile robots cooperate to transport a payload without grasp-points using non-prehensile manipulation. Because the end-effectors can only apply frictional pressure rather than adhere directly, DYNAMO

serves as a distributed analogue of dual arm manipulation, with each robot functioning as an independently mobile end-effector coordinated through shared contact and force sensing. To address mobile base pose tracking error and uncertainty during transport, a dynamic admittance parameterization strategy coupled with coordinated optimal planning is proposed.

Unlike traditional planning methods such as probabilistic road maps, which generate fixed paths based on environmental maps and assumed kinematics, this approach continuously adjusts the motion of each robot in response to real-time force feedback from the payload, enabling robust performance under uncertainty and unmodeled dynamics.

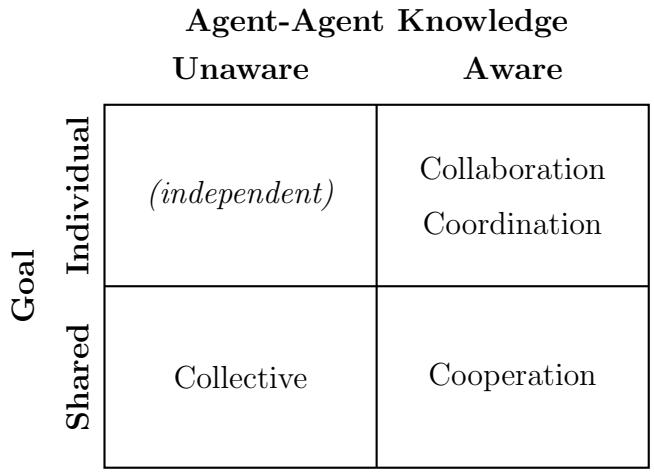


Figure 1.1: Types of interactions in multi-robot systems based on goal structure and agent awareness. Note that although collaboration and coordination share the same structure, the key difference is that in collaborative systems agent's actions benefit each other, while in coordinated systems agent's action cannot benefit from other agents [6].

.

The DYNAMO architecture comprises of two components, the optimal trajectory planner and the dynamic admittance parameterization controller, which fall into subtly different MRS paradigms. The nature of the trajectory planner is coordinated because each robot has their own trajectory to follow and doesn't influence the other robots actions. The nature of the manipulation component is cooperative because

the robots share a goal of maintaining sufficient friction force on the payload to hold it up. Figure 1.1 diagrams the subtle differences in the types of MRS paradigms including collective, collaborative, cooperative, and coordinated [6].

The DYNAMO architecture is validated on hardware with two JiANT robots. In experiments, two JiANT robots (see Figure 2.6) successfully transported a payload along a curved 2.2 m path in 100% of the 20 trials conducted using DYNAMO, whereas a position controller achieved 0% success rate and a static admittance controller achieved only 5 % to 25 % success rate. Compared to dynamic admittance parameterization, both position-based and static admittance control are observed to fail to adapt to these disturbances. The DYNAMO architecture is able to respond to contact forces and disturbances, compensating for misalignment and motion-induced errors.

1.1 Related Works

Cooperative transport is well studied in swarm robotics and animal behavior. Among animals, only ants and humans appear capable of cooperative lifting and stabilizing payloads through application of pressure by multiple individuals. Most other animal species employ relatively simple collective strategies such as pushing or dragging objects. See McCreery and Breed [7] and Berman et al. [8] for overviews of force negotiation and group dynamics in ants with application to group robotic transportation.

Our system avoids assumptions of perfect sensing or rigid couplings or grasping, unlike most prior work in simulation or fixed-base robot arms [9, 10, 11]. Gong et al. present a multi-robot system with novel six degree-of-freedom rigid connectors that enable hardware-based mechanical admittance [12]. Tuci et al. [13] classify cooperative transport as grasping, pushing, or caging. The DYNAMO architecture

is most closely related to caging.

The natural admittance controller introduced by Newman [14] models compliant behavior via mass-spring-damper systems. This foundation underlies the DYNAMO architecture. Methods for analytically designing admittance or impedance controllers include passivity methods, which can be leveraged for actively changing parameters to improve stability or controller convergence [15, 16].

For surveys of non-admittance control strategies, see [17]. Leader–follower approaches are reviewed in [18, 19]. Yufka and Ozkan [11] demonstrate a multi-robot system where the object is rigidly affixed to the robots, eliminating the need for compliance. In purely theoretical work, robot–payload teams are modeled as a unified system using Jacobians to optimize coordination [20, 21], assuming perfect knowledge of dynamics.

The state of the art in compliance-based multi-robot transport is work by Carey and Werfel [22] who simulate linear transport of a grasped payload using impedance control. In contrast, DYNAMO enables transport along curved trajectories with compliant, non-grasping end-effectors in hardware.

Admittance control has been applied in single-robot systems [23, 24], multi-arm settings [25], and simulations [26]. This thesis demonstrates its use in an adaptive tuning framework for multi-robot non-prehensile mobile transport using actual hardware.

Contributions

There are limitations and assumptions common in current MRSs capable of transportation. Rigid connectors are often employed to guarantee the robot maintains hold; however, many payloads and objects in general do not have the necessary ge-

ometric features or attachment points. Grasping manipulators are able to attach themselves to handles on the payload, but payloads with flat or curved surfaces, e.g. barrels, cannot be grasped at any point. Position-based control of the end-effector requires an extremely low margin of error and can be harmful to the manipulator or payload if disturbances lead to high forces. Lastly, although simulation is a common tool to develop and test robotic systems, it is not guaranteed to capture the complex dynamics and interactions that occur in reality. Hardware validation is a necessary step towards deploying algorithms in practical scenarios. The DYNAMO architecture is inspired by the limitations and assumptions in the aforementioned discussion. For clarity, the main contributions of this thesis are as follows.

- Custom non-prehensile end-effectors allow for the manipulation of irregular objects or objects without grasping points.
- A novel dynamic admittance parametrization module is presented which updates the parameters of admittance online in response to measured error.
- An optimal coordinated planner generates a trajectory for the mobile bases that minimizes snap and angular momentum along the path.
- Hardware experimentation and validation on two JiANT robots suggest DYNAMO performs better at transportation under conditions with high pose error.

Together, robust multi-robot transportation is enabled by dynamic admittance parametrization and optimal coordinated planning with non-prehensile manipulation. DYNAMO is shown to be successful under high pose error compared to position-based and static admittance methods.

The paper is organized as follows. Chapter 2 discusses the creation of the JiANT robot. Chapter 3 covers the challenges and features developed in software to operate

the JiANT. Chapter 4 proposes the DYNAMO architecture. Chapter 5 presents results from hardware experimentation. A conclusion is draw in Chapter 6.

Chapter 2

Hardware Development

The multi-robot transportation task requires each agent in the system to meet a minimum set of capabilities. For example, to implement admittance control there must be hardware that allows for the measurement of forces on the end-effector. To this effect, two JiANT robots were constructed to meet the hardware capabilities necessary to test the DYNAMO architecture. The JiANT robot is a modified Swarmie, fitted with commercially available and off-the-shelf components. The Swarmie is a descendant of the *iANT* robots built to emulate ant foraging [27]. Approximately 100 Swarmies were produced and distributed to 20 universities to support the NASA Swarmathon competition. Each Swarmie is designed to be assembled by students and are built from low-cost commodity components and 3D printed parts. The JiANT continues in the spirit of the Swarmie, being a similarly low-cost and handmade platform.

The DYNAMO architecture asks for several requirements from each mobile manipulator. At a minimum, each JiANT needed the following hardware components:

- **Compute**: An on board PC capable of high-level robotics algorithm implementation, running the robot operating system version 2 (ROS2) [28], and the

compute power for real-time high-frequency execution.

- **Manipulator**: A 4 degree-of-freedom manipulator to interact with the environment.
- **Plate End-Effector**: A plate end-effector for non-prehensile manipulation.
- **Force Sensor**: A force-sensing resistor (FSR) to measure forces at the port of interaction.
- **Low-Level Compute**: A small computing unit to interface with low-level hardware, i.e. command motors and read voltage passing through the FSR.
- **Vision**: Color and/or depth images to support localization and mapping.
- **Mounting Fixtures**: Carefully designed 3D structures to hold new components in place.

The challenges and motivations for each hardware modification from the original Swarmie are discussed in the following subsections.

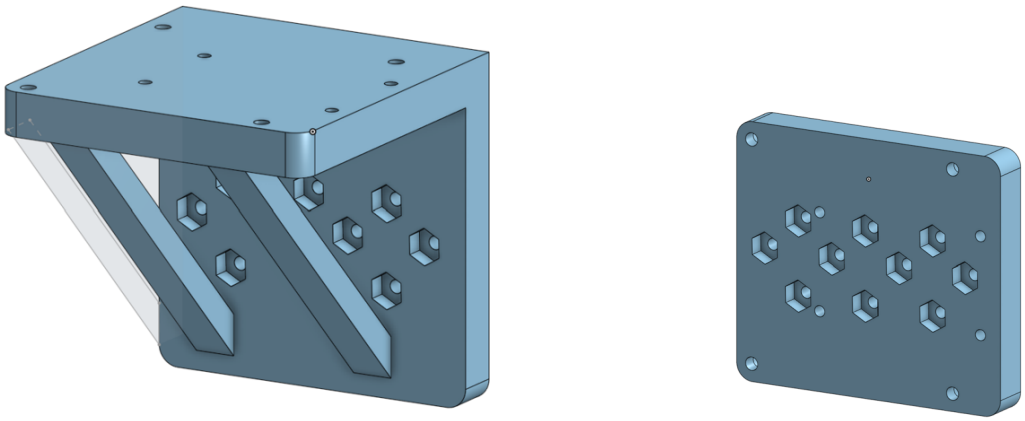
2.1 Compute

A full wireless mini PC capable of running the Ubuntu Jammy Jellyfish (22.04) operating system is necessary to run the ROS2 Humble framework, the robotics middleware of choice. An Intel NUC i7 mini PC was chosen for its light weight, small form, and reasonable computing power. This computer was placed on a 3D printed structure above the center of mass of the robot. It is powered by a 16 volt lipo battery inside the robot chassis prior to voltage regulation. The PC connects to both the PincherX 100 manipulator and the RealSense camera via USB 3.0 connectors.

2.2 Manipulator

A manipulator with sufficient degrees-of-freedom at the end-effector to accommodate uncertainty in base motion and localization is necessary for making contact with and exerting forces on the payload during transportation. The PincherX 100 manipulator arm supplied by Trossen Robotics was chosen for this purpose. The PincherX 100 is a 4 degree-of-freedom, 4R (revolute), serial manipulator. It is advertised to be capable of 300 mm reach, 50 g payload, 5 mm to 8 mm accuracy, and 5 mm repeatability. Notably, the PincherX 100 is fully integrated into the ROS2 Humble framework, which eases the software development workload.

The manipulator itself underwent several modifications while being integrated into the Swarmie platform. The base of the PincherX 100 was bulky and could not be attached to the Swarmie in any reasonable configuration. At first, the base of the PincherX 100 was removed and a custom flat plate joint was used to fix the manipulator to the Swarmie with a 90° roll in its orientation. The custom flat plate joint was modeled in house and printed using a 3D printer using polylactic acid (PLA) material (see Figure 2.1b). This mounting orientation for the manipulator was not natural and induced too much torque on the waist joint, as gravity was no longer acting on the manipulator's intended axis. In the second iteration, an angle joint (see Figure 2.1a) was modeled and printed to fix the manipulator to the Swarmie while maintaining its original upright orientation. In this orientation the manipulator was positioned with a 90° yaw rotation with respect to the mobile base. The manipulator's power cable was then integrated into the Swarmie's power bus and communicated with the computing unit via a USB 3.0 connection.



(a) Angle joint design

(b) Flat joint design

Figure 2.1: Joint designs for mounting the PincherX 100 manipulator to the Swarmie chassis. Two different approaches were modeled and 3D printed to evaluate mounting options.

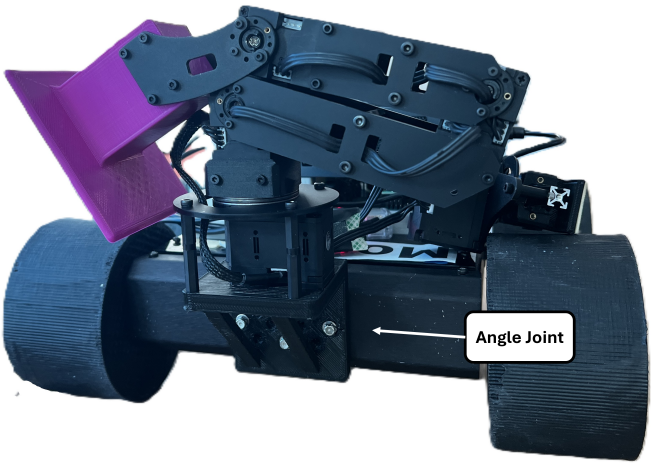


Figure 2.2: The PincherX 100 4 degree-of-freedom manipulator arm fixed to a JiANT using the angle joint modeled in Figure 2.1a. The manipulator is mounted on the front of the robot with a 90° yaw rotation with respect to the mobile base.

2.3 Plate End-Effector

A specialized end-effector is necessary to perform non-prehensile manipulation on the payload. A flat plate end-effector allows for controlled pushing over a large

area, as opposed to a finger-type end-effector that can only push on a precise point. The original gripper joint on the PincherX 100 was removed and replaced with a modeled and 3D printed flat plate end effector (see Figure 2.6). This custom part also reduced the weight of the end-effector, effectively increasing the capable payload of the manipulator.

2.4 Force Sensor

A force sensor capable of scalar force readings is necessary to produce force measurements for both admittance-based controllers. A 1.5 in square FSR 406 from Interlink Electronics was fixed to the flat plate end-effector. A small foam pad is applied to the surface of the FSR to evenly distribute force across the sensor (see Figure 2.6). The positive and negative connectors were then wired into a breadboard circuit that connected the FSR to a microcontroller in line with a 1 kΩ resistor (see Figure 2.3). This effectively mapped the 0 V to 5 V pin readings to real force values.

2.5 Low-Level Compute

A low-level microcontroller was needed to actuate the motors, read motor-encoder signals, and measure voltage going through the FSR. A Teensy 4.0 microcontroller was chosen for being easy to program, and compatible with micro-ROS software. The Teensy 4.0 was integrated into the breadboard and connected to the other low-level components (see Figure 2.3).

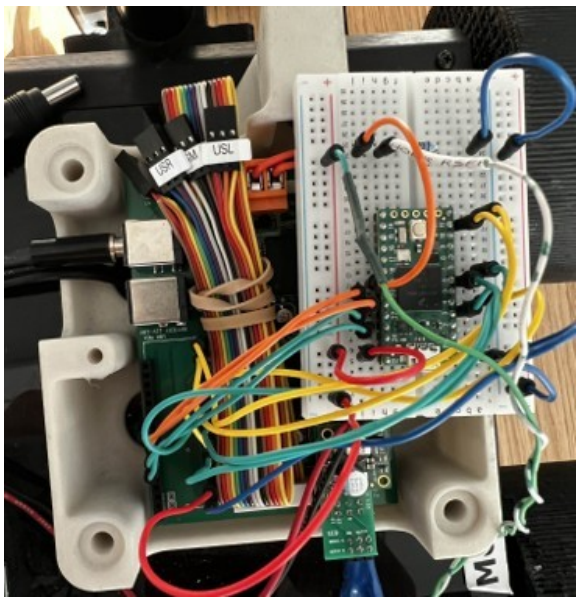


Figure 2.3: A photo of the breadboard configuration. The FSR leads, motor control signals, motor-encoder output signals, and Teensy 4.0 are all wired together.

2.6 Vision

A camera system is necessary for robot localization. An initial modification involved mounting a RealSense D455f to a carbon fiber mast that was fixed to the rear of the robot (see Figure 2.4). The sensor was situated such that it captured a wide field of view in front of the robot, specifically a horizontal field of view of 87° and vertical field of view of 58°. The RealSense D455f camera provides color images and depth images via stereo vision technology, utilizing dual infrared cameras and an infrared projector to calculate depth through stereo triangulation. The camera captures color images and depth images at a resolution of 1280×720 pixels and at a frame rate of 30 fps. The effective depth sensing range is 0.2 m to 6 m, making it a reliable choice for robot localization and environmental mapping. The camera also includes a built-in inertial measurement unit (IMU) with a 6-axis accelerometer and gyroscope, enabling visual-inertial odometry for improved localization accuracy.

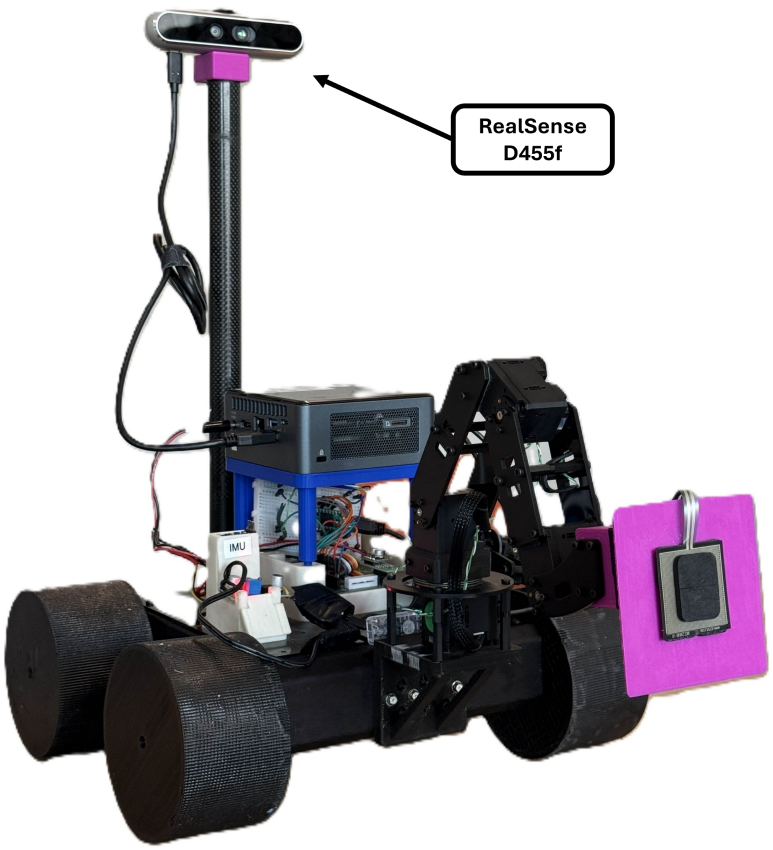


Figure 2.4: JiANT robot equipped with a RealSense D455f depth camera for environmental perception, providing synchronized color and depth data to enable obstacle detection, navigation, and 3D mapping.

However, the compute power needed to perform high-accuracy and high-frequency localization and mapping using depth images provided by the RealSense D455f was greater than the Intel NUC i7 could provide. As an alternative, a Vicon motion capture system was adopted to provide precise localization for the robots (see Figure 2.5). The Vicon system used sixteen Valkerie camera sensors positioned around a 4 m $\times$ 4 m workspace. The Vicon tracking system was configured to produce tracking data at a rate of 100 Hz. Vicon systems utilize passive retro-reflective markers, sometimes referred to as mocap markers, attached to both the robots and payload object. Mocap markers are then illuminated by infrared LEDs integrated into each

camera unit. Although the adopted Vicon system boasted sixteen Valkerie camera units, they were setup such that at any given moment approximately four cameras maintained line-of-sight to track each marked object. With even just four effective cameras, tracking estimation is still very accurate. The Vicon tracker software specifically provided six degree-of-freedom pose estimation (position and orientation) with sub-millimeter positional accuracy, significantly exceeding the precision and update rate achievable through the RealSense D455f camera paired with the Intel NUC i7 compute. This external localization approach freed up computational resources and also brought an opportunity for low error position tracking with the mobile bases. The RealSense camera and mast were removed in the final version of the robot seen in Figure 2.6.

2.7 Mounting Fixtures

Several mounting fixtures are necessary to securely hold all of the components in place during operation. The addition of the PincherX 100 manipulator arm motivated the creation of an angle joint depicted in Figure 2.1a used to attach the manipulator to the Swarmie. The high-level computing unit and low-level computing unit (Intel NUC and Teensy 4.0) are housed by a blue 3D printed scaffolding seen in Figure 2.6. Finally, to position the RealSense camera such that it has a clear forward view, a carbon fiber mast is attached to the base of the Swarmie for the camera to rest on top of (see Figure 2.4). These modifications were made to accommodate the hardware that was being tested at the moment.

Figure 2.5: A photo of the adopted Vicon camera system workspace and two JiANT robots. In the scaffolding are four Valkerie cameras. The Vicon system comprised of sixteen of these cameras. The pose estimation produced by the Vicon tracking software is transmitted over the same ROS2 network that connects the JiANT robots.

2.8 Final Robot Platform

The final robot platform is diagrammed in Figure 2.6. Two identical JiANT robots were built to develop and test the DYNAMO architecture.

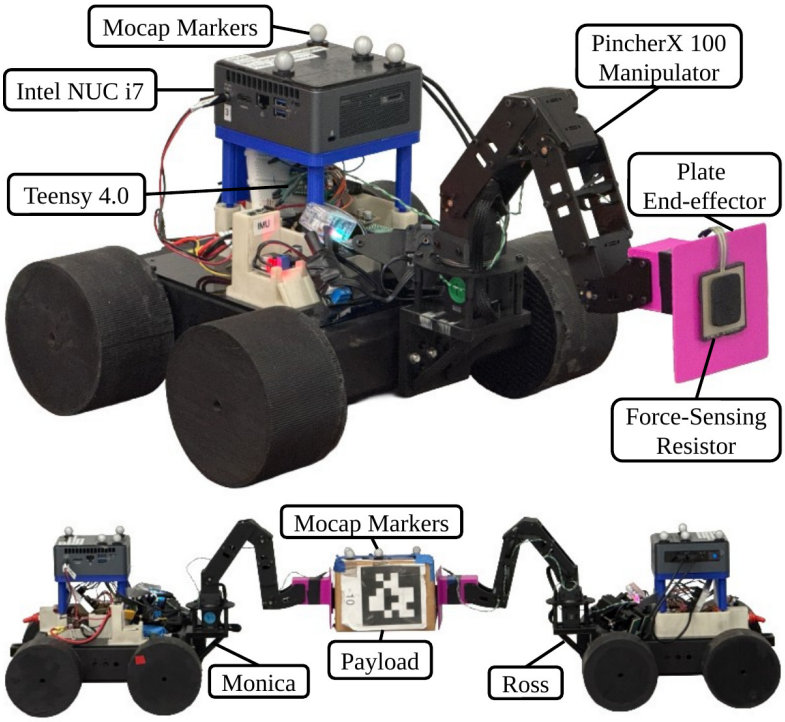


Figure 2.6: JiANT robots are modified Swarmie robots made from off-the-shelf and 3D printed components. Both Monica and Ross have identical hardware components and wiring.

Chapter 3

Software Development

Similar to the hardware design process, a software suite built primarily with the ROS2 Humble framework was implemented to meet the required software specifications and capabilities for the DYNAMO architecture. In addition to needing to communicate with various hardware components, it is crucial that the admittance and base tracking algorithms run at a high frequency. The following software capabilities were implemented on an Intel NUC i7 computer and Teensy 4.0 micro-controller:

- **Robotic Middleware**: Inter-process communication, message passing, and modular software architecture. ROS2 Humble provides the foundational framework for streamlined algorithm development.
- **End-Effector Actuation**: Real-time control and actuation of the robotic end-effector.
- **Force Estimation**: Calibration that maps sensor analog output to accurate force readings.
- **Localization and Mapping**: Global pose estimation and environmental mapping capabilities for autonomous localization and mapping.

- **Network Communication**: High-frequency, low-latency inter-device communication across the ROS2 network infrastructure.

The full implementation process for each software capability is discussed in the following subsections.

3.1 Robotic Middleware

A fundamental challenge of robotics is the coordination of diverse components such as sensors, actuators, and algorithms. Choosing a framework to facilitate that coordination is a critical first step. The most common approach, which is the one that was chosen, is to use ROS2, specifically the Humble distribution. ROS2 is an open-source, community-developed robotics middleware framework maintained by the Open Source Robotics Foundation (OSRF) that is designed to simplify and modulate the robot design process. ROS2 Humble, the most recent distribution at the time of development, was chosen for its vast array of modern developer tools, modular structure, and built-in communication infrastructure that supports multiple devices on a single network.

The most consistent hurdle throughout the development process was that most ROS2 tools and libraries were designed for single-robot systems. For example, the transform (TF) tree, a common interface for tracking and manipulating coordinate frames, expects the data to be published to the default TF topic. In a system with multiple robots, all topics are now name-spaced with the robot's name, which disrupts the default topic nomenclature. The only solution in this case and in many other cases was to carefully reroute topics or edit the source code to allow name-spaced topics. This kind of issue was frequent and recurring throughout the development of the DYNAMO architecture across many kinds of ROS2 developer tools,

PincherX 100 interfaces, RealSense interfaces, and other open-source robotics packages.

Another facet of ROS2 development is to create a complete description, or model, of the robot often referred to as the robot description. A robot description is typically a compilation of unified robot description format (URDF) and XML macro (XACRO) files. The robot description for the JiANT is shown in Figure 3.1. Since the JiANT robot is custom in-house constructed robot, the robot description was manually created using measurements from the actual robot.

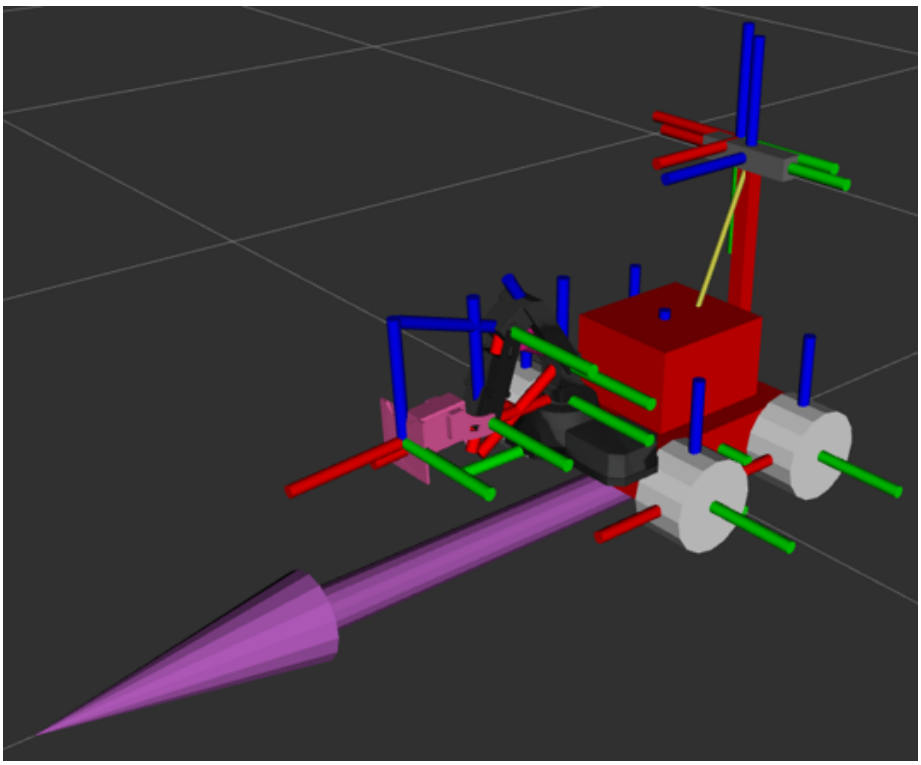


Figure 3.1: A visualization of the robot description for a JiANT. This model is used in many core packages of ROS2.

3.2 End-Effector Actuation

Control of the end-effector is enabled by solving the inverse kinematics (IK) problem. IK solutions typically come in numerical or analytical forms. An analytical solution involves solving for the exact solution through closed-form equations based on the robot's geometric structure, however, this formulation can be complex and yield multiple solutions for redundant manipulators. A numerical solution iteratively converges towards a solution until the end effector pose is within a certain tolerance. For real-time, complex manipulators, as is this case, a numerical IK solver is typically chosen. The MoveIt2 framework and the Interbotix API were tested for end-effector actuation.

MoveIt 2 is a ROS2-based framework for robotic manipulation and motion planning. It provides high-level functionality that ties together kinematics, collision checking, motion planning, and control profiles. MoveIt2 features several different libraries for motion planning. Two libraries were tested within this framework. These were called the Open Motion Planning Library (OMPL) and the Pilz Industrial Motion Planner (PILZ).

OMPL provides a collection of sampling-based motion planning algorithms including Rapidly-exploring Random Tree (RRT), Rapidly-exploring Random Tree Connect (RRTConnect), and Kinematic Planning by Interior-Exterior Cell Exploration (KPIECE). These algorithms are well suited for high dimensional configuration spaces. Several tests were performed with this library and it was found to be highly unsuitable for DYNAMO. PILZ is another motion planner available in MoveIt 2. It was originally developed for industrial robots, and provides deterministic, Cartesian and joint-space planners designed for predictable, safe, and certifiable paths. The trajectories generated from these motion planners were more reasonable than the ones from OMPL. Table 3.1 provides a summary on the speed of the discussed

planners.

Table 3.1: Comparison of MoveIt 2 path planners.

Path Planner	**Avg. Planning Time (s)**
RRT	0.36
RRTConnect	0.32
SBL	0.15
EST	0.14
BKPiece	0.12
LBKPiece	0.25
PILZ LIN Path Planner	0.05

The next library utilized was the Interbotix API, a software interface developed by the manufacturer of the PincherX 100 manipulator. This API provided a high level abstraction layer over the servo motors. It consisted of functions written in Python that would take a goal pose, then use a numerical IK solver to find the corresponding joints angles. This library provided greater freedom and control over the arm and allowed for the development of a custom position controller to generate trajectories.

An initial implementation of custom position control of the end-effector using the Interbotix API involved an iterative update strategy. This position controller acted as a layer on top of the IK solver, continuously reading in goal poses and adjusting the position of the arm in small increments. At each iteration, the controller compared the arm's current pose with the given target pose. If the error was above a threshold, an intermediate target pose was generated that nudged the arm toward the target. This intermediate target pose was then passed to the IK solver, which generated a new set of joint angles. The servo motor drivers would receive these joint angles, and autonomously move the motors into these positions. This process repeated continuously.

This iterative position control method allowed for continuous corrections, but

introduced instability when combined with the admittance controller. Furthermore, the arm velocity was very slow for small displacements under the servo's default velocity profiles. The position controller was redesigned and the velocity profile of the servo motors were changed to "step" to address these issues. All velocity profiles are shown in Figure 3.2. The "step" velocity profile allowed the servo motors to achieve their commanded position as quickly as possible, regardless of the distance.

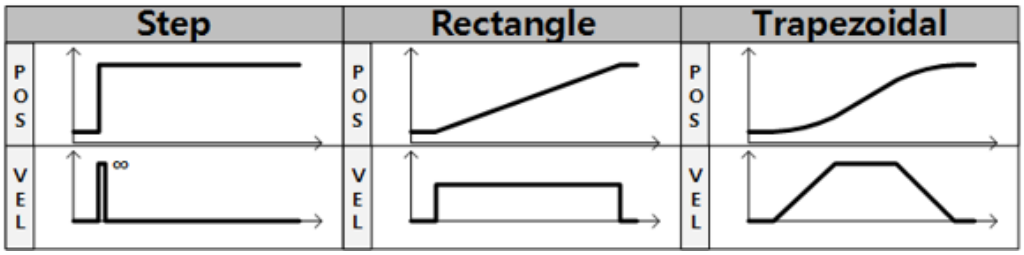


Figure 3.2: The PincherX 100 manipulator arm actuates its servos using one of three possible velocity profiles: step, rectangle, or trapezoidal [29]. The default is trapezoidal.

The manufacturer's default configuration was a time-based trapezoidal profile, diagrammed in Figure 3.2, which constrained the motion to gradual accelerations proportional to the distance to the target [29]. This conservative approach ensured smooth motion but resulted in slow response times, particularly for small movements. The velocity profile was disabled, meaning the velocity of the servo motors would resemble a step function. In this modified scheme, the actuators moved at the maximum feasible velocity until the target position was reached. This adjustment significantly improved responsiveness while maintaining sufficient stability to operate in conjunction with the admittance controller.

The final design of the admittance controller removed the iterative position control approach and opted to go straight to the target pose. On each iteration of the control loop, if the error is above a threshold, the arm moves to that position directly instead of to an intermediate target position. This was done to avoid the destabilizing and jerky effects seen in the iterative method.

3.3 Force Estimation

An essential component to the DYNAMO architecture is the force estimation provided by the FSR. A 1.5 in square FSR 406 from Interlink Electronics uses a simple conversion between force and voltage. The voltage divider is described as:

$$V_{out} = \frac{R_M V+}{(R_M + R_{\text{FSR}})} \tag{3.1}$$

where R_m represents the measuring resistor, and the value of R_m affects the sensitivity range of the FSR. R_m was tested using 22 kΩ, 47 kΩ, 100 kΩ, 220 kΩ, and 470 kΩ resistors. For each resistor, 10 samples were taken using 5 g, 10 g, 25 g, 50 g ,100 g, 200 g, and 500 g weights. For each dataset collected, a log curve was fitted and graphed in Figure 3.3.

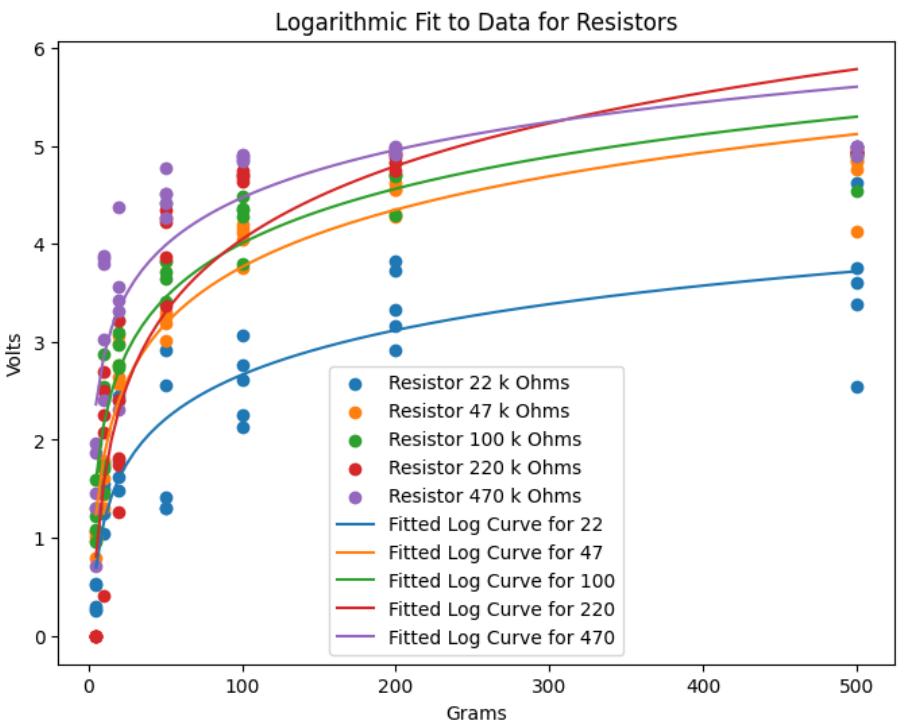


Figure 3.3: The output of the FSR was sampled several times using different weights for a set of resistors.

Table 3.2: Third-degree polynomial coefficients for force sensor calibration.

Coefficient	Value
a_3	0.035582
a_2	-0.055113
a_1	0.38334
a_0	-0.057718

The final curve used and implemented on the Teensy 4.0 was a third-degree polynomial of the form:

$$F = a_3x^3 + a_2x^2 + a_1x + a_0 \tag{3.2}$$

where F represents the calibrated force output, x is the raw analog sensor reading, and $a_0, a_1, \ldots, a_3$ are the polynomial coefficients listed in Table 3.2. This fitted curve has an R value of

3.4 Localization and Mapping

As a prerequisite to reliable navigation, the robot must first localize itself by determining its pose within the environment. Robot localization comprises of two approaches: local estimation, in which the robot estimates its current pose relative to its previous known pose, and global localization, in which the robot determines its pose relative to the global environment, typically through a global mapping algorithm. Local estimation often feeds into a global estimation algorithm for increased reliability. This section outlines challenges faced when obtaining reliable odometry, as well as development experiences with the Simultaneous Localization and Mapping (SLAM) Toolbox and Real-Time Appearance-Based Mapping (RTAB-Map) [30, 31].

To first establish a strong local pose estimation method, an extended Kalman filter (EKF) [32] comprised of IMU data and wheel odometry data was developed. IMU data originated from the RealSense camera, while wheel odometry was calculated on

the Teensy 4.0 microcontroller using forward kinematics (FK) on the output from the motor encoders with the robot description. Together in an EKF, local estimation was found to be decent but far too much error accumulated after driving more than about 1.5 m for the admittance-based methods to compensate (see Figure 3.4). This motivated the implementation of a global pose estimation algorithm.

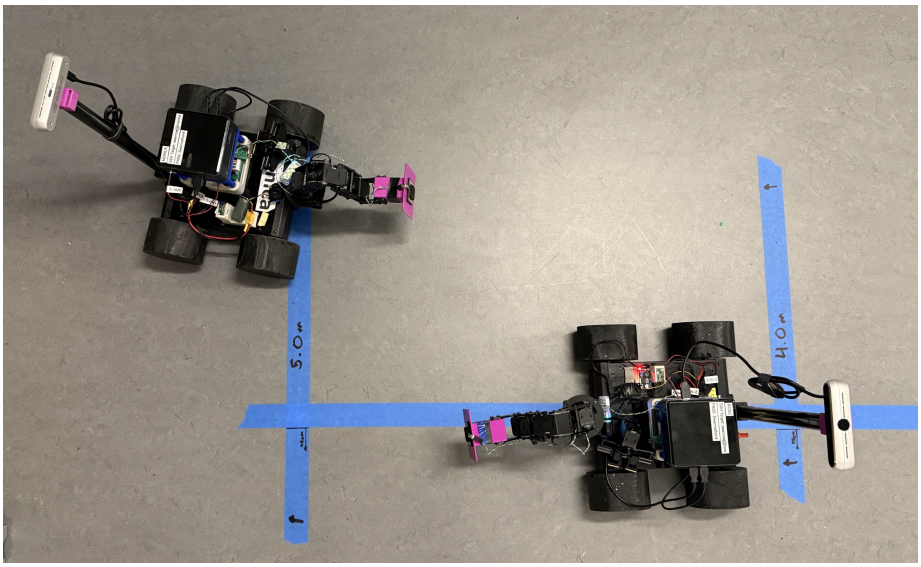


Figure 3.4: The result of both JiANTs after being commanded to drive straight 4 m. The robots are actuated using a proportional-integral-derivative (PID) controller that corrects for error in actual position returned by the EKF. In this case, although error is corrected for, the EKF does not have a reliable notion of where the robots are in the environment. This motivated the need for a global localization algorithm.

Our initial approach was to use the SLAM Toolbox together with ROS 2 Navigation Stack (Nav2) to both map and navigate the robot's environment. While the maps generated by the SLAM Toolbox were acceptable, the localization quality was very poor. Hoping to achieve more accurate pose estimates, RTAB-Map was tested, only to discover that the root cause of poor estimation quality lay in the quality of the onboard sensors and insufficient compute to run these algorithms at a high resolution.

RTAB-Map, another SLAM package available in ROS2, was relatively easy to

configure but ultimately did not produce more reliable pose estimates. In many cases, it struggled to generate a consistent, usable map of the environment, often producing warped or incomplete reconstructions. After experimenting with several configurations, it became clear that the robot's sensors produced too much noise to be trustworthy.

The biggest problem came from the wheel encoders and the IMU sensor on the RealSense camera. Granted that the robot is skid-steer, the wheels frequently slipped on the ground, causing the robot to move in ways impossible for the encoders to capture. The IMU sensor added further complications, with excessive noise and inaccuracies of up to 15 degrees. Some configurations even caused systematic drift, where RTAB-Map believed the robot was constantly turning despite being stationary. It is important to recognize that SLAM and RTAB-Map are impressive algorithms but also require lots of compute and tuning to perform optimally. The challenge of optimally tuning a global pose estimator was outside of the research scope.

3.5 Navigation

The method of navigation is a core component of the cooperative transportation process. Although an admittance-based controller is able to adapt to changes in relative distance between the robots, they are limited by the reach and dynamics of the manipulator. A well designed navigator will produce a trajectory that not only minimizes the error in the distance between the robots but also minimizes higher order terms such as snap and angular momentum.

An initial approach for developing a trajectory planner was inspired by Yukfa et al. [11]. In this work, a group of nonholonomic agents (designated "followers") and the payload (designated the "virtual leader") fall into a formation. The virtual leader charts a path to the goal that minimizes sharp turns so that the nonholonomic bases

can track the correct motion. This work served as a good example of multi-agent navigation, and introduced the nuances of multi-robot path planning. Ultimately, the decision was made to implement a custom navigation algorithm.

The next attempt at a more sophisticated navigation strategy was to use cubic splines to form a trajectory, and proportional-integral-derivative (PID) controllers to get the robots to follow this trajectory. First define two points, a starting point and a final point. Then, intermediate points are added to form a non-linear path. After the path is formed, the robots would start to follow their own trajectories independently. Each robot used three PID controllers to stay on track. One PID controller simply controlled the x-position of the robot. The other two worked together to control the heading of the robot while also minimizing its lateral error. This cascaded PID architecture is described in depth in Section 4.2.1. This strategy was simple, but had two limitations. First, the cubic splines introduced jerk at the beginning of the path, which caused the robots to immediately drop the payload. Second, there was no reliable way to get the robots to form and follow the same path. Usually, the path one robot would follow would be a slightly shifted from the path of the other robot. Furthermore, there were no guarantees in relative position or coordination between the mobile bases since each spline was generated independently.

The last path planning strategy implemented, and the one ultimately used in the final product, is a sequential convex quadratic program (SCQP) optimal coordinated trajectory planner. This planner minimizes snap and change of angular momentum along a coordinated payload path for two nonholonomic robots. This algorithm is described in full detail in Section 4.2.

3.6 Network Communication

Throughout the duration of the project, ROS2 communication was facilitated via Wi-Fi using a consumer-grade router as the central access point. While this configuration initially proved sufficient bandwidth, the network infrastructure began exhibiting significant performance degradation as the software stack grew in complexity and computational demand.

The first major bottleneck was observed while testing the admittance controller on both robots. Specifically, starting the software stack on a second robot consistently resulted in a collapse of network throughput, effectively halting all inter-process communication. To mitigate this issue, the underlying ROS2 middleware was changed from FastDDS to CycloneDDS. This immediately yielded a noticeable improvement in discovery speed and message throughput. However, this modification served only as a temporary solution. Once SLAM and Nav2 were integrated into the software stack, the same network communication failures resurfaced. A more robust solution was achieved by giving a ROS2 domain ID to each robot to isolate their network traffic, effectively creating isolated communication channels for each robot.

After transitioning to a Vicon system, communication failures once again became a recurring problem. When both robots were initialized, the network exhibited the same collapse as before, halting data exchange and compromising experimental integrity. To address this, the quality of service settings across all topics were modified to best effort reliability and volatile durability. This prevented communication from halting. However, this came at the cost of message reliability. Many critical messages were either dropped or never received. Reducing the publishing frequency of high-bandwidth topics further improved stability but failed to eliminate packet loss entirely, leaving the system only partially functional. This issue continually persisted into the final research product and remains a problem.

Chapter 4

Methods

The proposed DYNAMO architecture is implemented by deriving the relevant control theory, developing supporting software, and constructing a robotic system capable of executing the controller and planning a path for the mobile base. This thesis describes the construction of JiANTs and the experimental design used to evaluate DYNAMO's performance in physical trials. Throughout the paper, all numerical values, including robot specifications and experimental results, are reported to two significant figures unless otherwise noted. The robots that are experimented with are named Monica and Ross in reference to a TV show in which a group of friends struggle to cooperatively carry a couch.

4.1 Control Methods

4.1.1 Static Admittance Control

An admittance controller is defined by the relationship between force and motion at the "port of interaction" [33]. The port of interaction is the point at which the

manipulator interacts with the environment. By modeling the compliant mechanical behavior of a mass-spring-damper system, forces at the end-effector are translated into smooth motion trajectories. This can prevent the external forces from either damaging the manipulator's hardware or the payload itself. The general formula for admittance control in one-dimensional free space is given in Equation (4.1) where F_{ext} is the external force, M is the mass, B is the damping, K is the stiffness, and x is the relative distance from the virtual position.

$$F_{\text{ext}} = M\ddot{x} + B\dot{x} + Kx \tag{4.1}$$

A frame of reference is introduced by defining a virtual set point x_v as shown in Equation (4.2) where x is the relative distance from x_v. The virtual set point defines the position of the end-effector when $F_{\text{ext}} = 0$. By setting x_v to a position that is inside of the payload, a controlled force, F_{ext}, can be exerted.

$$F_{\text{ext}} = M\ddot{x} + B\dot{x} + K(x_v - x) \tag{4.2}$$

Initial velocity and acceleration of the system are set to zero, i.e. $\dot{x} = 0$ and $\ddot{x} = 0$. A step size of $\Delta t = 0.002$ is set to reflect the desired 500 Hz update rate. Then Euler's method is used to compute the position $(x_v - x_t)$ generated by the admittance controller in Equation (4.5) [23].

$$\ddot{x}_t = \frac{1}{M}(F_{\text{ext}} - B\dot{x}_t - K(x_v - x_t)) \tag{4.3}$$

$$\dot{x}_t = \dot{x}_{t-1} + \ddot{x}_t \Delta t \tag{4.4}$$

$$x_t = x_{t-1} + \dot{x}_t \Delta t \tag{4.5}$$

Admittance is implemented on each of the the JiANT robots, modeling each as a mass-spring-damper system on either side of the payload as seen in Figure 4.1a. In practice, this architecture allows each manipulator to support the the payload and cooperatively regulate forces at their respective end-effector.

The parameters for the static admittance controller (Table 4.1) were initially chosen using the critical damping heuristic. Specifically, the damping was set above the critical value $B_{\text{crit}} = 2\sqrt{MK}$ to ensure a stable, fast, non-oscillatory response. These values were then hand tuned over several weeks to maintain consistent contact between each manipulator and the payload prior to driving. Values were also tuned such that the forces on the end effector mapped to positions within the manipulators range of motion.

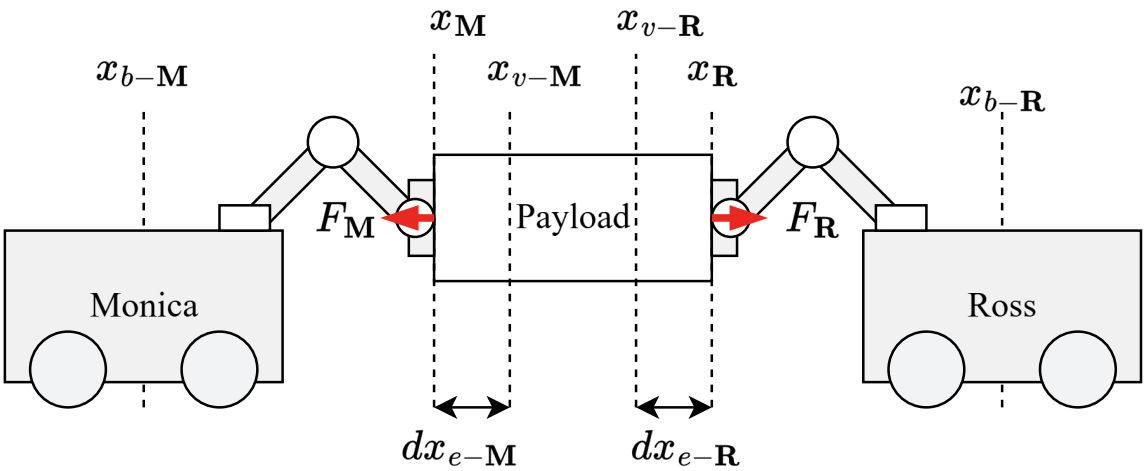
4.1.2 Dynamic Parameterization of Admittance-Based Control

An admittance controller such as the one implemented in Section 4.1.1 is parameterized (i.e. tuned) for a specific range of forces at the port of interaction and assuming a fixed distance. However, during transport the distance, d, between mobile manipulators varies continuously. An admittance controller tuned for the initial state before movement begins may not work well once transportation is underway. The parameters set for the static controller (Table 4.1) are used as initial values for the dynamic version.

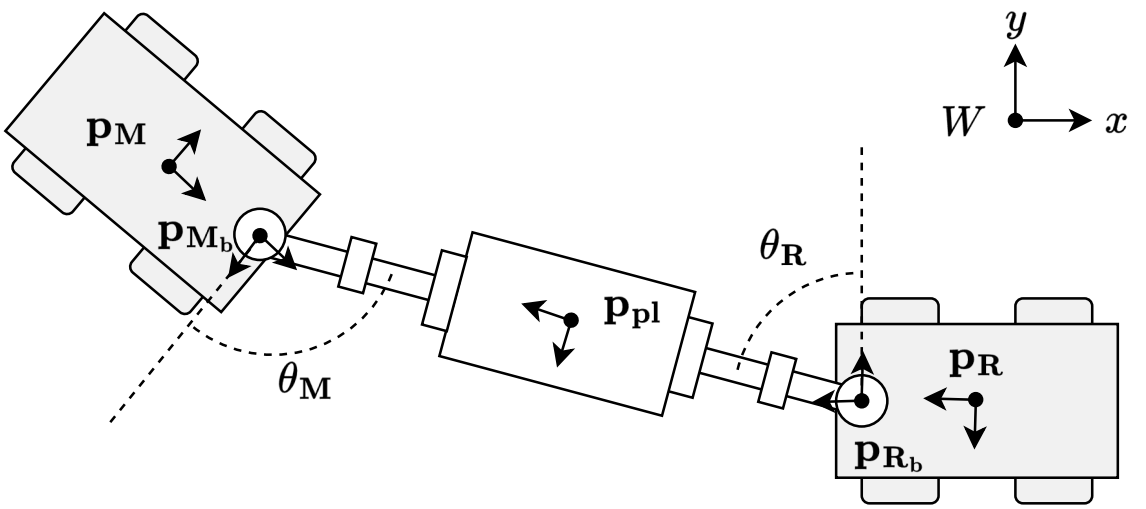
Table 4.1: Dynamic admittance controller parameters. Default values were empirically determined under the constraint that they satisfy the critical-damping heuristic. These default values were used for the static admittance controller and as initial values for the dynamic admittance parametrization.

Symbol	Meaning	Value	Units
x_v	Default virtual position	0.25	m
K	Default stiffness	60	N/m
B	Default damping	70	N·s/m
M	Default mass	5	kg
α_x	Gain for x_v update	0.5	dimensionless
α_K	Gain for K update	30	dimensionless

A dynamically parameterized admittance controller that updates K and x_v during



(a)



(b)

Figure 4.1: Mechanical diagrams of the system: (a) Lateral view of the transportation setup. (b) View from above of the payload, mobile base, and manipulator poses for each robot labeled $\mathbf{p_{pl}}$, $\mathbf{p_M}$, $\mathbf{p_R}$, $\mathbf{p_M}$ respectively. The adjusted heading angles $\theta_{\mathbf{M}}$ and $\theta_{\mathbf{R}}$ are calculated using the aforementioned poses in Section 4.1.3 to ensure the manipulator poses are always collinear along the x-axis.

transportation based on the distance between the two robots is proposed. These parameters are updated via the estimation algorithm shown in Figure 4.2 (yellow). The update rule for the virtual set pose, x_v', and the stiffness, K', is defined as follows:

$$x_v' = x_v + \alpha_x e \tag{4.6}$$

$$K' = K - \alpha_K e \tag{4.7}$$

where $e = |\mathbf{p}_M - \mathbf{p}_R|$ is the magnitude of the error in relative position between Monica and Ross, x_v is the default parameter for virtual position, K is the default parameter for stiffness, α_x is the scaling factor for virtual pose, and α_K is the scaling factor for stiffness. After modifying Equation (4.2) to use the adaptive versions x_v' and K', the dynamic admittance controller in Equation (4.8) is produced.

$$\begin{aligned} F_{\text{ext}} &= M\ddot{x} + B\dot{x} + K'(x_v' - x) \\ F_{\text{ext}} &= M\ddot{x} + B\dot{x} + (K - \alpha_K e)((x_v + \alpha_x e) - x) \end{aligned} \tag{4.8}$$

Using Euler's method as described in Equation (4.5), together with the formula for dynamic admittance from Equation (4.8), and solving for the displacement $(x_t - x_v)$ yields Equation (4.9),

$$\begin{aligned} x_v - x_t = ((x_v + \alpha_x e) - x_{t-1}) + \Delta t(\dot{x}_{t-1} \\ + \Delta t\left(\frac{1}{M}(F_{\text{ext}} - B\dot{x} - (K - \alpha_K e)x_{t-1})\right). \end{aligned} \tag{4.9}$$

The update rules defined in Equations (4.6) and (4.7) seek to ensure safety, stability and robustness at the port of interaction. The gain α_x ensures kinematic consistency by formulating a virtual pose that tracks payload-relative motion. The gain α_K ensures force regulation as a function of inter-robot spacing by updating effective stiffness. When the mobile bases converge, the manipulators are at risk of mechanical interference. To discourage this motion, stiffness is increased to prevent collision. Conversely, when the mobile bases diverge, the payload is at increased risk

of being dropped, and the manipulators extend according to Equation (4.6). In this lengthened configuration, manipulability at the end-effector is decreased, and motion requires greater torque on the joints. Lowering effective stiffness reduces transmitted force on the payload while mitigating the risk of exceeding manipulator joint torque limits.

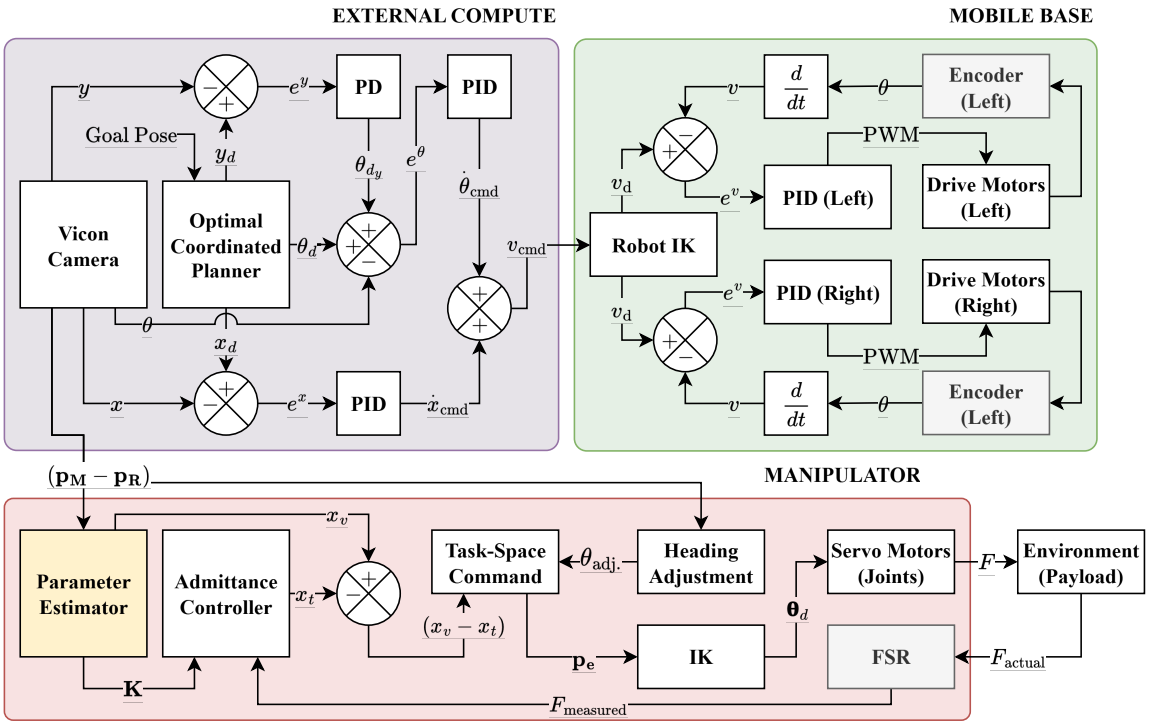


Figure 4.2: A block diagram for a single agent. The parameter estimation algorithm, highlighted in yellow, takes as input the displacement in the relative position of both JiANTs while the mobile base executes the desired trajectory from the optimal coordinated planner. The only external signals received by each robot are the actual robots poses, $[x, y, \theta]$, and command velocities.

4.1.3 Manipulator Heading

The dynamic admittance controller adjusts the target position x along a single axis given F_{ext}, a scalar force measurement. The single-axis constraint requires end-

effector poses to remain collinear within the 2D special Euclidean group (SE(2)), meaning their positions and orientations $\langle x, y, \theta \rangle$ must be aligned along a shared axis. To guarantee this behavior the necessary heading angle adjustment $\theta_{\mathbf{M}}$ and $\theta_{\mathbf{R}}$ are calculated, diagrammed in Figure 4.1b. Let $\mathbf{p_M}$ and $\mathbf{p_R}$ be the mobile base poses with respect to the world frame, W, for Monica and Ross. Let $\mathbf{p_{M_b}}$ and $\mathbf{p_{R_b}}$ be the pose of the manipulator base link for each robot. The transformations $T_{W,\mathbf{M}}$ and $T_{W,\mathbf{R}}$, from W to $\mathbf{p_M}$ and W to $\mathbf{p_R}$ respectively, are dynamically retrieved via the Vicon camera system. The static transformations $T_{\mathbf{p_M},\mathbf{p_{M_b}}}$ and $T_{\mathbf{p_R},\mathbf{p_{R_b}}}$ from the mobile base to the manipulator base link are fixed measured transformations.

Frame composition is used to calculate $T_{\mathbf{p_{M_b}},\mathbf{p_{R_b}}}$, the transformation from one robot manipulator to the other, and *vice versa* as described in Equations (4.10) and (4.11).

$$T_{\mathbf{p_{M_b}},\mathbf{p_{R_b}}} = T^{-1}_{\mathbf{p_M},\mathbf{p_{M_b}}}\, T^{-1}_{W,\mathbf{M}}\, T_{W,\mathbf{R}}\, T_{\mathbf{p_R},\mathbf{p_{R_b}}} \tag{4.10}$$

$$T_{\mathbf{p_{R_b}},\mathbf{p_{M_b}}} = T^{-1}_{\mathbf{p_R},\mathbf{p_{R_b}}}\, T^{-1}_{W,\mathbf{R}}\, T_{W,\mathbf{M}}\, T_{\mathbf{p_M},\mathbf{p_{M_b}}} \tag{4.11}$$

The desired rotation for each manipulator arm is determined via $\text{atan2}(y, x)$, where y and x are planar components of the translation vector $\mathbf{p} = [x\ y\ z]^\top$ in the homogeneous transformation. Specifically, the last column of $T_{\mathbf{p_{M_b}},\mathbf{p_{R_b}}}$ for Monica and $T_{\mathbf{p_{R_b}},\mathbf{p_{M_b}}}$ for Ross is extracted from the transformation.

In practice, this alignment is maintained by coordination over a network between the JiANTs using the Vicon for accurate localization. Although the JiANTs align their manipulator arms using globally localised pose information, the core cooperative control strategy, dynamic admittance control, is decentralised and relies solely on local force sensing. This distinction is important: communication is used for pose alignment and mobile base localisation, not for synchronising or planning manipulation forces. The reactive behaviour of each robot is governed independently by sensed interaction forces at the end effector.

4.1.4 Position Controlled Transportation

A static position controller for the end-effector acts as a baseline for measuring the value of admittance control. In this control scheme, the end-effector maintains a fixed relative position located slightly inside the payload. For successful transportation, the JiANTs must maintain a fixed separation distance.

4.2 Optimal Coordinated Planning

Successful mobile manipulation requires coordinated path planning for both agents. To do this, cooperative transport is parameterized by considering the pose and dynamics of the payload $\mathbf{p_{pl}}(t) = [x(t), y(t), \phi]^\top \in \mathbb{R}^3$, following Slightam et al. [34]. The unit vectors defining the payload pose are

$$\hat{\mathbf{v}}(\phi) = \begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix}, \hat{\mathbf{n}}(\phi) = \begin{bmatrix} -\sin\phi \\ \cos\phi \end{bmatrix}, r = \tfrac{L}{2} + a. \tag{4.12}$$

where r is the payload pose offset, $\hat{\mathbf{v}}$ is the unit tangent/forward direction of the payload (body x-axis) and $\hat{\mathbf{n}}$ is the unit normal/left direction (body y-axis) of the payload. A subset of $\mathbf{p_{pl}}$, to denote the x,y, positions is $\mathbf{c} = [x(t), y(t)]^T$. This can be used to define the base positions and velocities of the robots,

$$\mathbf{b}_1 = \mathbf{c} - r\hat{\mathbf{v}}, \qquad \mathbf{b}_2 = \mathbf{c} + r\hat{\mathbf{v}}, \tag{4.13}$$

$$\dot{\mathbf{b}}_1 = \dot{\mathbf{c}} - r\,\dot{\phi}\,\hat{\mathbf{n}}, \qquad \dot{\mathbf{b}}_2 = \dot{\mathbf{c}} + r\,\dot{\phi}\,\hat{\mathbf{n}}. \tag{4.14}$$

To determine the skid–steer platform headings (assuming no lateral slip), first define

$$\theta_i = \mathrm{atan2}((\dot{\mathbf{b}}_i)_y, (\dot{\mathbf{b}}_i)_x), \tag{4.15}$$

where θ_i are the base headings for Monica and Ross. With the geometry defined, the optimization problem can be setup to co–optimize 4 different components of the multi-agent motion. (1) minimum snap of the centerline: penalize $\|\mathbf{c}^{(4)}\|^2$. (2) minimum change of angular momentum: for a planar rigid body, $L_z = I_z\dot{\phi} \Rightarrow \dot{L}_z = I_z\ddot{\phi}$; penalize $\dot{L}_z^2 = I_z^2\ddot{\phi}^2$. (3) Soft yaw–tangent alignment: discourage large misalignment between box yaw ϕ and the tangent heading $\psi = \text{atan2}(\dot{y}, \dot{x})$. Finally, (4) Tracking/boundary terms: keep the center-line close to a desired S-path between the start location and desired goal location and enforce boundary conditions.

The minimization function, J, is initially written for a strictly convex quadratic program (QP) as:

$$\begin{aligned} \min_{\mathbf{X},\mathbf{Y},\mathbf{\Phi}} \quad J = {} & w_{\text{track}}\big(\|\mathbf{X} - \mathbf{X}^{\text{ref}}\|_2^2 + \|\mathbf{Y} - \mathbf{Y}^{\text{ref}}\|_2^2\big) \\ & + w_{\text{snap}}\big(\|D_4\mathbf{X}\|_2^2 + \|D_4\mathbf{Y}\|_2^2\big) \\ & + w_{\dot{L}}\,\|D_2\mathbf{\Phi}\|_2^2 + w_{\ddot{L}}\,\|D_3\mathbf{\Phi}\|_2^2 \end{aligned} \tag{4.16}$$

D_i are co-state variables and the yaw–tangent coupling is left out initially. The Hessian is symmetric positive definite (SPD), so Equation (4.16) solves in $\mathcal{O}(N)$ with a band solver. The term, w_{snap}, limits center-line jerkiness; $w_{\dot{L}}$ penalizes torque effort (since $\tau_z = \dot{L}_z = I_z\ddot{\phi}$); $w_{\ddot{L}}$ limits torque *rate*.

The constraints for the yaw to follow the tangent heading are $\psi = \text{atan2}(\dot{y}, \dot{x})$, but ψ is nonconvex in $(\mathbf{X}, \mathbf{Y})$. A quadratic *alignment* term is added and linearized about the previous iterate.

To achieve coupling in sequence, the linearization of the heading map is required, by letting $\dot{x}_k = (D_1\mathbf{X})_k,\ \dot{y}_k = (D_1\mathbf{Y})_k,\ s_k^2 = \dot{x}_k^2 + \dot{y}_k^2,\ \psi_k = \text{atan2}(\dot{y}_k, \dot{x}_k)$. Its first variation is

$$\delta\psi_k = \frac{\partial\psi_k}{\partial\dot{x}_k}\delta\dot{x}_k + \frac{\partial\psi_k}{\partial\dot{y}_k}\delta\dot{y}_k = \Big(-\frac{\dot{y}_k}{s_k^2}\Big)\delta\dot{x}_k + \Big(\frac{\dot{x}_k}{s_k^2}\Big)\delta\dot{y}_k.$$

Since $\delta\dot{x} = D_1\,\delta\mathbf{X}$, $\delta\dot{y} = D_1\,\delta\mathbf{Y}$, the linearized alignment residual at iterate j is

$$\begin{aligned} r^{(j)} = &\underbrace{(\mathbf{\Phi} - \mathbf{\Phi}^{(j)})}_{\text{yaw increment}} + \underbrace{(\mathbf{\Phi}^{(j)} - \mathbf{\Psi}^{(j)})}_{\text{constant}} \\ &\underbrace{- P_x^{(j)} D_1(\mathbf{X} - \mathbf{X}^{(j)}) + P_y^{(j)} D_1(\mathbf{Y} - \mathbf{Y}^{(j)})}_{\text{tangent increment}}, \end{aligned} \tag{4.17}$$

with diagonal gains $P_x^{(j)} = \operatorname{diag}\big(-\dot{y}_k^{(j)}/s_k^{(j)2}\big)$, $P_y^{(j)} = \operatorname{diag}\big(\dot{x}_k^{(j)}/s_k^{(j)2}\big)$, and $\mathbf{\Psi}^{(j)} = [\psi_k^{(j)}]$. Then $w_{\text{align}}\|r^{(j)}\|_2^2$ is added to the cost. This produces only quadratic (and banded) cross terms between $\mathbf{X}, \mathbf{Y}, \mathbf{\Phi}$.

Equation (4.18) extends the cost function defined in Equation (4.16) by adding an alignment term. At each iteration j, solve the convex QP,

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{\Phi}} \; J(\mathbf{X}, \mathbf{Y}, \mathbf{\Phi}) + w_{\text{align}}\|r^{(j)}(\mathbf{X}, \mathbf{Y}, \mathbf{\Phi})\|_2^2, \tag{4.18}$$

subject to boundary equalities (position, and zero end–velocities/yaw–rates) and payload bounds. Following this, $(\mathbf{X}^{(j+1)}, \mathbf{Y}^{(j+1)}, \mathbf{\Phi}^{(j+1)}) \leftarrow (\mathbf{X}, \mathbf{Y}, \mathbf{\Phi})$ is set and repeated until convergence occurs (small change of J or of the variables).

Once $(\mathbf{X}, \mathbf{Y}, \mathbf{\Phi})$ are found, iteratively determine the terms in Equations (4.12) to (4.15), described via Algorithm 1 using a SCQP.

4.2.1 Algorithmic Control Architecture

Desired positions, $\mathbf{b}_{1,k}, \mathbf{b}_{2,k}$, and associated headings, $\theta_{1,k}, \theta_{2,k}$, at step k for each mobile base determined by Algorithm 1 are recovered via Equations (4.13) and (4.15). Simultaneously, the measured positions $\hat{\mathbf{b}}_{1,k}$, $\hat{\mathbf{b}}_{2,k}$, $\hat{\theta}_{1,k}$, and $\hat{\theta}_{2,k}$ are retrieved from the Vicon system. Typical of skid-steer mobile platforms, a PID controller maps longitudinal error to linear command velocities, and a cascaded PD-PID controller maps lateral and heading error to angular command velocities as diagrammed in Figure 4.2.

Algorithm 1 SCQP for snap+$\dot{L}$ co–optimization

Build S–path samples $\mathbf{X}^{\text{ref}}, \mathbf{Y}^{\text{ref}}$ & initial $\mathbf{\Phi}^{(0)}$
Initialize $\mathbf{X}^{(0)} = \mathbf{X}^{\text{ref}}$, $\mathbf{Y}^{(0)} = \mathbf{Y}^{\text{ref}}$.
for $j = 0, 1, 2, \ldots$ **do**
    Compute $\dot{x}^{(j)} = D_1\mathbf{X}^{(j)}$, $\dot{y}^{(j)} = D_1\mathbf{Y}^{(j)}$, $s^{(j)} = \sqrt{\dot{x}^{(j)2} + \dot{y}^{(j)2}}$, $\mathbf{\Psi}^{(j)} = \text{atan2}(\dot{y}^{(j)}, \dot{x}^{(j)})$.
    Form $P_x^{(j)} = \text{diag}(-\dot{y}^{(j)}/s^{(j)2})$, $P_y^{(j)} = \text{diag}(\dot{x}^{(j)}/s^{(j)2})$.
    Solve the convex QP 4.18 with boundary equalities (banded SPD).
    Update $\mathbf{X}^{(j+1)}, \mathbf{Y}^{(j+1)}, \mathbf{\Phi}^{(j+1)} \leftarrow \mathbf{X}, \mathbf{Y}, \mathbf{\Phi}$.
    if $J^{(j)} - J^{(j+1)} < \varepsilon$ and $\|(\mathbf{X}^{(j+1)}, \mathbf{Y}^{(j+1)}, \mathbf{\Phi}^{(j+1)}) - (\mathbf{X}^{(j)}, \mathbf{Y}^{(j)}, \mathbf{\Phi}^{(j)})\| < \varepsilon$ **then break**
    end if
end for
Recover $\mathbf{b}_{1,2}, \dot{\mathbf{b}}_{1,2}, \theta_{1,2}$ via Equations (4.13) to (4.15).

The desired and actual positions are decomposed $\mathbf{b}_{i,k} = [x_{i,k}, y_{i,k}]^\top$ and then transformed into their respective mobile base frame using the transformations $T_{\mathcal{WM}}$ and $T_{\mathcal{WR}}$. For longitudinal correction, the error between the desired and actual x position in the mobile base frame $e^x_{i,k} = x_{i,k} - \hat{x}_{i,k}$ is passed into a PID controller which produces $\dot{x}_{\text{cmd}}$, a linear command velocity. For lateral and heading correction, a cascaded controller combines these errors into a single angular command velocity about the z-axis. Lateral error in the mobile base frame, $e^y_{i,k} = y_{i,k} - \hat{y}_{i,k}$, is passed through a PD controller which produces a $\theta_{d,k}$ correction. Then error $e^\theta_{i,k} = \theta_{d,k} + \theta_{i,k} - \hat{\theta}_{i,k}$ is passed through a PID controller which produces $\dot{\theta}_{\text{cmd}}$, an angular command velocity about the z-axis.

4.3 Experimental Methods

4.3.1 Hardware Implementation

The JiANT (Figure 2.6) robots used to conduct experiments are modified *Swarmies* themselves descended from *iANT* robots built to emulate ant foraging [27]. Approximately 100 Swarmies were produced and distributed to 20 universities. Swarmies are designed to be assembled by students and are built from low-cost commodity components and 3D printed parts. JiANTs continue in that spirit with the whole arm and sensor assembly costing under US$1000. Each robot uses a Pololu brushed DC motor to move the wheels. These motors are controlled using a Teensy 4.0 programmed using Arduino and micro-ROS [35]. An Intel NUC mounted on the JiANT runs ROS2 nodes. Swarmies are skid-steer robots. The JiANT base is 24 cm long and 33 cm wide.

An Interbotix PincherX-100, a 4DOF robotic arm with a reach of 30 cm, and accuracy of 8 mm was attached to the base. The arm is constructed from lightweight aluminum and actuated by Dynamixel servos. The arm is rated for payloads up to 50 g. The arm has a 30 cm maximum extension. This assembly was selected for its low cost and compatibility with ROS2.

The pincher was replaced with a 3D printed flat-plate end-effector equipped with a FSR. The FSR selected is a low-cost, consumer-grade product 38×38 mm from Interlink Electronics. This polymer-film sensor is 0.5 mm thick and lightweight (1.2 g), allowing integration into the end effector with minimal additional mass. The stock FSR can detect loads from approximately 50 g to approximately 10 kg. To improve sensitivity in the low-force range, the FSR was paired with a fixed 1 kΩ resistor in a voltage divider configuration. This FSR measures external force, F_{ext}, in the admittance equations in Section 4.1.1.

The payload is a small untreated cardboard box of mass 86.64 g (Figure 2.6). A thin foam pad covers the end-effector pressure sensor to distribute force and improve readings. The dynamics of cardboard–foam friction is complex. Preliminary experimentation showed even slight pressure deviations caused the box to fall despite contact friction.

4.3.2 Experimental Setup

Three different control strategies: admittance control (Section 4.1.1), dynamic parameterization (Section 4.1.2), and position control (Section 4.1.4) were tested under the same conditions to measure their effectiveness. All three strategies used the base navigation described in Section 4.2. Each strategy was evaluated using a set of 20 trials, where the only difference was the manipulator control strategy. In each trial, the robots were commanded to transport the payload along a path to a goal destination.

A Vicon tracking system was used to track the pose of each robot and the payload. Each robot's base and the payload had 4 markers attached. 4 Vicon cameras were able to effectively capture a 3×3 m area. Since the robots start off about 1 m apart that limited the maximum path distance to 2.2 m or approximately 9 robot base lengths.

To evaluate how well the payload could be transported under linear and curved movement, a goal location was chosen that resulted in the planner generating an S-shaped path as shown in Figure 4.3. The goal position was chosen to be to the right of the start location for half the trials and to the left for half the trials. This mirroring introduced variety in trajectory direction while keeping the overall movement pattern comparable across trials.

Experiments were conducted until 20 valid trials had been collected per strategy. Results are presented as distance traveled (Figure 5.1a) and success rate (Fig-

ure 5.1b). Trials affected by hardware and software issues (e.g. failed servos and ROS2 initialization problems) unrelated to the strategy being tested were excluded. These failures constituted less than 10% of total trials. For each trial, a payload was placed between the robots so the manipulators made contact. The trajectory planner was launched and the JiANTs navigated to the goal. Each trial is classified as a success if the payload is transported without being dropped for the entire distance of the planned path. If the payload is dropped, the integrated path distance covered before failure is recorded. The point at which the payload drops is recorded on videos of each trial and by observing forces applied to the end effectors. Successful trials are classified as "perfect" or "faulty". Faulty trials indicate that the payload shifted or twisted visibly during transport indicating an insecure hold on the payload.

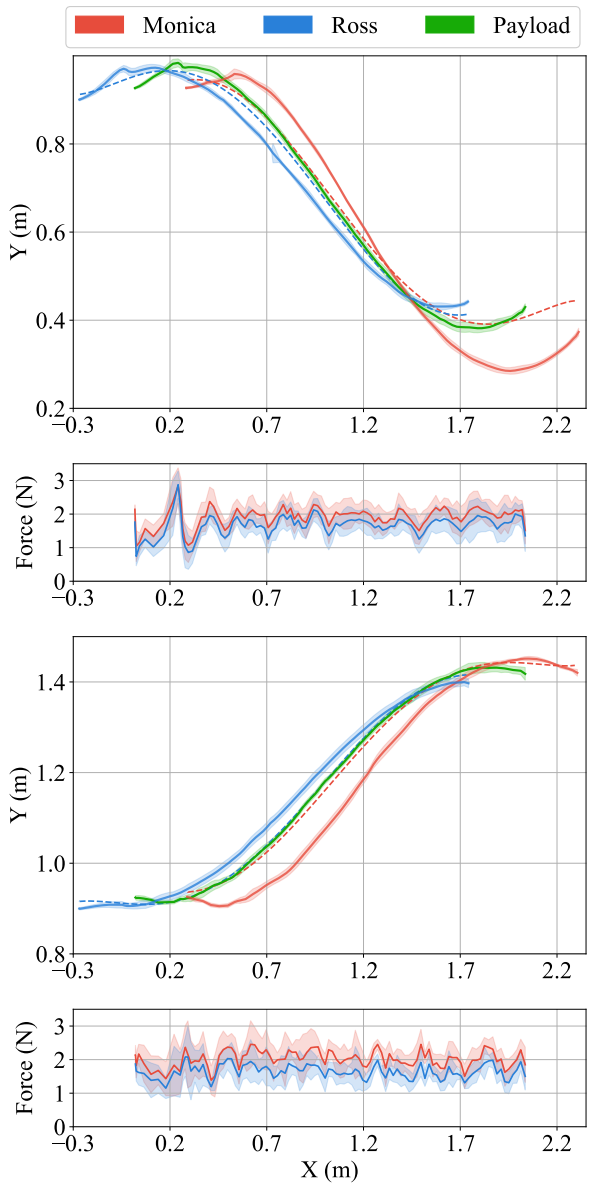


Figure 4.3: Motion along the desired trajectory and accompanying forces on the end-effector for dynamic admittance parameterization along a left s-curve and a right s-curve. The trajectory plots show the desired position (dashed line), mean position (solid line), and standard deviation (shaded area). The force plots, aligned with trajectory along the x-axis, show the mean force, μ, (solid line) and standard deviation, σ (shaded area). Despite error tracking the desired trajectory, forces are maintained throughout transportation.

Chapter 5

Results

Figure 4.3 shows the trajectory and interaction forces recorded during payload transport under the dynamic admittance controller. Although the applied forces exhibit some fluctuation, they appear stable, never exceeding the mechanical limits of the manipulators, based on qualitative experimental interactions. Notably, the robots consistently apply moderate, well-regulated contact forces that keep the payload secure throughout the motion. This suggests that the dynamic admittance controller actively modulates compliance to accommodate unmodeled dynamics and mobile base errors.

The effectiveness of this control strategy is evident in Figure 5.1. Figure 5.1a shows the distance traveled before the payload was dropped using each strategy. The dynamic admittance controller consistently reached the full 2.2 m extent of the trajectory, limited only by the Vicon workspace. By contrast, the position and static admittance controllers failed to reach even half that distance on average. This may be due to high tracking errors seen in Figure 4.3.

Across $n = 20$ experiments for each method, the position controller had a mean transport distance of 0.134 m (95% CI [0.112 m, 0.157 m]), the static admittance

controller 0.955 m (95% CI [0.515 m, 1.396 m]), and the dynamic admittance parameterization controller 2.185 m (limited by the arena size). Dynamic admittance achieved a significantly greater transport distance than both position control (mean difference 2.05 m, Welch's t-test, $p \ll 0.001$) and static admittance (mean difference 1.23 m, $p < 0.001$). Static admittance also exceeded position control (mean difference 0.82 m, $p < 0.001$).

As can be seen in Figure 5.1b, the dynamic admittance controller achieved a 100% success rate, completing all 20 trials without dropping the payload. In contrast, the static admittance controller completed only one successful trial, and the position controller failed in all attempts. This is expected with high tracking error as the position controller does not update the pose of its end-effector. The static admittance controller completed 4 faulty trials. Trials are considered to be either successful, faulty, or a failure. A faulty trial is one in which the payload does not fall to the ground, but a stable grasp is not maintained either. This means the payload may slip or rotate while being held, a behavior that isn't ideal for transportation.

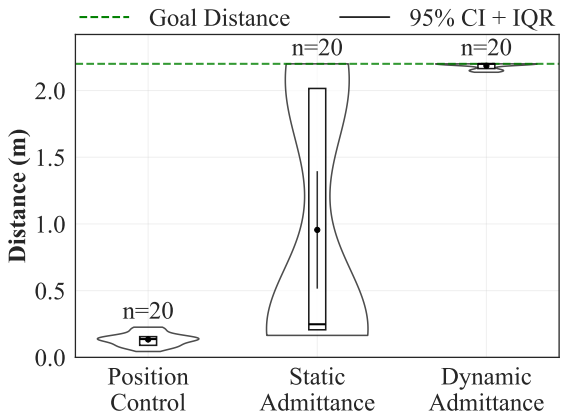
Taken together, these results confirm that the dynamic admittance controller maintains safe force levels, adapts to disturbances, and achieves reliable transport, in stark contrast to the failures observed under the other control strategies.

Experimental data is available online.[1] as is the code used to implement the position, admittance, and DYNAMO controllers.

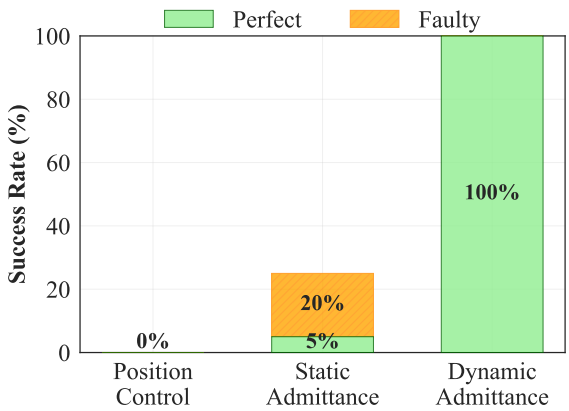
5.1 Discussion

The success of DYNAMO observed through experimentation likely arises from its ability to modulate compliance in real-time based on sensed interaction forces. This

[1] https://github.com/BCLab-UNM/MARIAM



(a) Violin plot of payload transport distances under three control strategies. Boxes show inter-quartile range; vertical black lines indicate 95 % confidence intervals for the median. Closed circles denote medians: 0.13 m (position), 0.21 m (admittance), and 2.18 m (dynamic admittance). Trials were capped at 2.2 m due to Vicon workspace limits.



(b) Bar chart of trial outcomes across control types. Green bars show percentage of successful trials; orange hatched bars indicate faulty trials. Dynamic admittance control yields the highest success rate.

Figure 5.1: Performance comparison across three control strategies for cooperative payload transport.

adaptation enables it to maintain stable end-effector contact with the payload, despite variations in base alignment, accelerations and snap, and unmodeled physical effects. The force traces in Figure 4.3 show that the controller is actively regulating interaction for the entire duration of transport. It maintains applied forces in

a bounded, effective range, avoiding both excessive pressure that could destabilize or damage the system and insufficient contact that would drop the payload. Simultaneously, the optimal coordinated trajectory plots a course for both nonholonomic bases. However, the error accumulated by the JiANT robots while tracking that trajectory places offline methods such as static admittance and position-based control at a disadvantage. Under a maximum pose error of about 10 cm it is expected that static admittance control and position-based control will fail to compete with dynamic admittance parameterization. This potential flaw with the experiment results ultimately highlights the benefits of the proposed work and applicability in real-world applications.

The static admittance controller offered limited improvement, completing only one successful trial and 5 faulty trials. Its inability to handle high pose error in the mobile base may have stemmed from its fixed admittance parameters, which cannot adapt to changing conditions or external disturbances. Static admittance does not update its parameters in response to mobile base errors like dynamic admittance parametrization, but it still operates a method to regulate forces and the end effector. There may be a set of admittance parameters such that static admittance leads to successful transportation 100% of the time. This set of ideal parameters would need to be carefully tuned for the specific manipulator, payload, and expected range of disturbances experiences along transportation. A major limitation of this formulation is that static admittance is not adaptable across robot robot systems. The specific tuning of one system may not guarantee the same success rate on different machines or in different conditions. Even under ideal tuning, static admittance is not as adaptable as dynamic admittance parameterization.

In contrast, the position controller failed to transport the payload entirely. The controller provided no compensation via the arms to correct for accumulated or sudden base displacement. As expected, even minor amounts of error between the

mobile bases would result in payload loss from lack of contact. Although position control has no ability to account for mobile base pose error, in ideal condition it may be possible for position-based successful transportation. As seen in related works, position-based control can the transportation of a payload in simulation under perfect pose measurement and actuation [10]. In reality, there will always be some amount of actuation error stemming from both the manipulator and the mobile base. Furthermore, a position-controller will never regulate forces at the port of interaction. In position-based control, the manipulator effectively behaves as a rigid object. This type of interaction is potentially harmful to both the manipulator and the payload. Under rigid control, inevitable disturbances in the system can lead to unexpected and high forces. The manipulator's joints may be damaged or overloaded. The payload, depending on its material, could be at risk of being damaged or destroyed. Admittance, which modulates these forces at the port of interaction, serves as a safety mechanism even under conditions with perfect pose.

The dynamic admittance parameterization controller overcame these shortcomings by using feedback to adjust interaction behavior on the fly. This has several advantages. First, it allows the system to respond to conditions that are difficult to model, such as unexpected interactions between the wheels and floor. Second, it avoids the need for exact identification of the payload's inertial or frictional properties. Third, it is generalizable to new hardware, tasks, or robot configurations, making it especially suitable for decentralized and heterogeneous multi-robot systems.

More broadly, these results highlight the value of adaptive compliance in distributed cooperative manipulation. By tuning admittance in response to sensed error, the system remains robust to disturbances without requiring explicit coordination or communication of internal states, making the approach both scalable and tolerant to partial failures. In parallel, successful mobile manipulation requires

coordinated path planning for both agents. To achieve this, we used an optimal cooperative transport planner which considers the pose and dynamics of the payload, enabling the generation of dynamically feasible trajectories that account for coupled motion. Together, the optimal navigation planner and the dynamic admittance parameterization controller provide a practical solution to non-prehensile multi-robot manipulation tasks in which grasping the payload is not possible.

Chapter 6

Conclusion

This thesis explores the performance of dynamic admittance parameterization with optimal coordinated planning on the task of cooperative robotic transportation. In the DYNAMO architecture, the optimal coordinated planning module produces an idealized trajectory for the robots to follow while the dynamic admittance parameterization module allows for the manipulators to adjust for disturbances from the environment and errors accumulated in the mobile bases. This combination of coordinated planning and modulation of forces at the port of interaction is demonstrated to correct for large disturbances and error during transportation. The use of admittance-based manipulation, which regulates forces on the end-effector, encourages safe actuation of the manipulator's joints and allows for the transportation of fragile or delicate payloads. Non-prehensile manipulation enables the transportation of irregular payloads or payloads without grasping points. All together, this research demonstrates the potential for admittance-based non-prehensile MRSs to solve the general task of transportation on a wide variety of payloads.

The DYNAMO architecture was implemented in hardware on JiANT mobile manipulators with odometry data sourced from a Vicon camera system. Experiments

were conducted to compare the success rate of DYNAMO against a statically parameterized admittance controller and a position-based controller. The experimental results demonstrate that under optimal coordinated planning, dynamic admittance parameterization achieves 100% task success ($n = 20$ trials) compared to 5% ($n = 20$ trials) for static admittance and 0% ($n = 20$ trials) for position control. The experimental results suggest that the DYNAMO architecture's ability to regulate force at the end effector enables safe and stable interaction with the environment. It also demonstrates that dynamic parameterization is able to account for large margins of error and disturbances from the mobile bases. This method of online force modulation and optimal coordinated planning may be beneficial for successful transportation in general.

It is important to acknowledge that the experimental results do not fully explore the complexities and nuances of cooperative transportation. Although this thesis validates the success of DYNAMO on hardware, experimentation is limited to two low-curvature short-distance paths. Transportation can involve much more complicated navigation challenges, e.g. long distances, object avoidance, changing velocity profiles, etc. In addition, DYNAMO was validated on mobile platforms that reached upwards of 10 cm in pose error, a difference that put the position-based controller and static admittance controller at a greater disadvantage.

The accuracy of a mobile platform can depend on the quality of the hardware, working conditions, and control algorithms implemented. A MRS may be deployed in environments of varying difficulty, with higher or lower guarantees on the margin of error for mobile base pose. A transportation system operating outside on rough terrain would have to accommodate higher errors and greater disturbances. While transportation in warehouse conditions may utilize motion capture technology such as Vicon and operate under a much smaller margin of error. DYNAMO shows proficiency under high operating errors, but can also guarantee force regulation in low

error conditions. This wide range of capability makes DYNAMO a more adaptable system for the varying challenges and obstacles associated with mobile coordinated transportation. Furthermore, the DYNAMO architecture paves the way for MRS architectures capable of address the general task of transportation for varied object types in a safe and robust manner.

The experimental results suggest that dynamic admittance parameterization with optimal coordinated planning performs better in terms of success rate and stable force regulation for the task of cooperative transportation. Future work intends to cover the limitations of the experiments section and strengthen the argument for dynamic admittance parameterization for cooperative transportation tasks in general. The first item of improvement would be to validate the success rate of dynamic admittance parametrization and methods it is compared against under much lower tracking error. Future work would also further explore the cooperative transportation task space by experimenting with different types of trajectories, different velocity profiles, and different types of mobile platforms.

References

[1] Avinash Gautam and Sudeept Mohan. A review of research in multi-robot systems. In *2012 IEEE 7th International Conference on Industrial and Information Systems (ICIIS)*, pages 1–5, 2012.

[2] Jaydeep Kshirsagar, Sam Shue, and James M. Conrad. A survey of implementation of multi-robot simultaneous localization and mapping. In *SoutheastCon 2018*, pages 1–7, 2018.

[3] Sarah Ackerman, George Fricke, Joshua Hecker, Kastro Hamed, Samantha Fowler, Antonio Griego, Jarett Jones, Jake Nichol, Kurt Leucht, and Melanie Moses. The swarmathon: An autonomous swarm robotics competition. *arXiv preprint arXiv:1805.08320*, 05 2018.

[4] Francois Bonnet and Xavier Defago. Exploration and surveillance in multi-robots networks. In *2011 Second International Conference on Networking and Computing*, pages 342–344, 2011.

[5] Hamed Farivarnejad and Spring Berman. Multirobot control strategies for collective transport. *Annual Review of Control, Robotics, and Autonomous Systems*, 5(1):205–219, 2022.

[6] Lynne E. Parker. Distributed intelligence: Overview of the field and its application in multi-robot systems. *Journal of Physical Agents*, 2(1):5–14, 2008.

[7] Helen F McCreery and MD Breed. Cooperative transport in ants: a review of proximate mechanisms. *Insectes sociaux*, 61(2):99–110, 2014.

[8] Spring Berman, Quentin Lindsey, Mahmut Selman Sakar, Vijay Kumar, and Stephen C. Pratt. Experimental study and modeling of group retrieval in ants as an approach to collective transport in swarm robotic systems. *Proceedings of the IEEE*, 99(9):1470–1481, 2011.

[9] B. Hichri, J-C. Fauroux, L. Adouane, I. Doroftei, and Y. Mezouar. Design of cooperative mobile robots for co-manipulation and transportation tasks. *Robotics and Computer-Integrated Manufacturing*, 57:412–421, 2019.

[10] Zuguang Liu and Md Suruz Miah. Cooperative Object Transportation Using Autonomous Networked Cobots: A Distributed Approach. In *2024 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, pages 1–7, Chemnitz, Germany, June 2024. IEEE.

[11] Alpaslan Yufka and Metin Ozkan. Formation-based control scheme for cooperative transportation by multiple mobile robots. *International Journal of Advanced Robotic Systems*, 12(9):120, 2015.

[12] Quan Liu Zhao Gong, Zhenguo Nie and Xin-Jun Liu. Design and control of a multi-mobile-robot cooperative transport system based on a novel six degree-of-freedom connector. *ISA Transactions*, 139:606–620, 2023.

[13] Elio Tuci, Muhanad HM Alkilabi, and Otar Akanyeti. Cooperative object transport in multi-robot systems: A review of the state-of-the-art. *Frontiers in Robotics and AI*, 5:59, 2018.

[14] W. S. Newman. Stability and performance limits of interaction controllers. *Journal of Dynamic Systems, Measurement, and Control*, 114(4):563–570, 12 1992.

[15] Jonathon E. Slightam, Daniel R. McArthur, Steven J. Spencer, and Stephen P. Buerger. Passivity analysis of quadrotor aircraft for physical interactions. In *2021 Aerial Robotic Systems Physically Interacting with the Environment (AIRPHARO)*. IEEE, 2021.

[16] Jonathon E. Slightam and Antonio D. Griego. Deep neural network design for improving stability and transient behavior in impedance control applications. In *2023 ASME/IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. ASME/IEEE, 2023.

[17] Abdollah Amirkhani and Amir Hossein Barshooi. Consensus in multi-agent systems: a review. *Artificial Intelligence Review*, 55(5):3897–3935, 2022.

[18] Jinyan Shao, Guangming Xie, Junzhi Yu, and Long Wang. Leader-Following Formation Control of Multiple Mobile Robots. In *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.*, pages 808–813. IEEE, 2005.

[19] Xing An, Celimuge Wu, Yangfei Lin, Min Lin, Tsutomu Yoshinaga, and Yusheng Ji. Multi-robot systems and cooperative object transport: Communications, platforms, and challenges. *IEEE Open Journal of the Computer Society*, 4:23–36, 2023.

[20] Jie Chen and Shixiong Kai. Cooperative transportation control of multiple mobile manipulators through distributed optimization. *Science China Information Sciences*, 61(12), December 2018.

[21] Dongdong Qin, Jinhui Wu, Andong Liu, Wen-An Zhang, and Li Yu. Cooperation and coordination transportation for nonholonomic mobile manipulators: A distributed model predictive control approach. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 53(2):848–860, 2022.

[22] Nicole E Carey and Justin Werfel. Collective transport of unconstrained objects via implicit coordination and adaptive compliance. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12603–12609. IEEE, 2021.

[23] Daniel R. McArthur, Jonathon E. Slightam, Steven J. Spencer, and Stephen P. Buerger. Coordinated admittance-impedance control with force excitation for compliant, underactuated aerial manipulation. *IFAC-PapersOnLine*, 58(28):558–563, 2024. The 4th Modeling, Estimation, and Control Conference – 2024.

[24] Hossein Gholampour, Jonathon E. Slightam, and Logan Beaver. Mass-adaptive admittance control for robotic manipulators. In *2025 Modeling, Estimation and Control Conference (MECC)*. IFAC, 2025.

[25] Yong Li, Chenguang Yang, Weisheng Yan, Rongxin Cui, and Andy Annamalai. Admittance-based adaptive cooperative control for multiple manipulators with output constraints. *IEEE transactions on neural networks and learning systems*, 30(12):3621–3632, 2019.

[26] Shraddha Barawkar, Mohammadreza Radmanesh, Manish Kumar, and Kelly Cohen. Admittance Based Force Control for Collaborative Transportation of a Common Payload Using Two UAVs. In *Volume 3: Vibration in Mechanical Systems*, page V003T39A007, Tysons, Virginia, USA, October 2017. American Society of Mechanical Engineers.

[27] Joshua P Hecker and Melanie E Moses. Beyond pheromones: evolving error-tolerant, flexible, and scalable ant-inspired robot swarms. *Swarm Intelligence*, 9(1):43–70, 2015.

[28] Steven Macenski, Tully Foote, Brian Gerkey, Chris Lalancette, and William Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science robotics*, 7(66):eabm6074, 2022.

[29] ROBOTIS. *DYNAMIXEL XL430-W250*. ROBOTIS Co., Ltd., 2024.

[30] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE Robotics & Automation Magazine*, 13(2):99–110, 2006.

[31] Mathieu Labbé and François Michaud. RTAB-Map as an Open-Source Lidar and Visual Simultaneous Localization and Mapping Library for Large-Scale and Long-Term Online Operation. *Journal of Field Robotics*, 36, 2019.

[32] Thomas Moore and Daniel W. Stouch. A generalized extended kalman filter implementation for the robot operating system. In *Annual Meeting of the IEEE Industry Applications Society*, 2014.

[33] Neville Hogan and Stephen Buerger. Impedance and Interaction Control. In Thomas Kurfess, editor, *Robotics and Automation Handbook*. CRC Press, October 2004.

[34] Jonathon E. Slightam, Andrew J. Steyer, Logan E. Beaver, and Carol C. Young. An approach to realize generalized optimal motion primitives using physics informed neural networks. *Letters in Dynamic Systems and Control*, 5(2), 2025.

[35] Kaiwalya Belsare, Antonio Cuadros Rodriguez, Pablo Garrido Sánchez, Juanjo Hierro, Tomasz Kołcon, Ralph Lange, Ingo Lütkebohle, Alexandre Malki, Jaime Martin Losa, Francisco Melendez, et al. Micro-ros. In *Robot Operating System (ROS) The Complete Reference (Volume 7)*, pages 3–55. Springer, 2023.