Improving Robotic Lego Assembly with Vibro-Tactile Feedback

Chris Chang

CMU-RI-TR-24-41

July 24, 2024

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Submitted in partial fulfillment of the requirements for the degree of Masters of Science in Robotics.

"Education is the only thing you always bring with you."
Thank you Mom and Dad for teaching me what's important.

Abstract

Robotic manipulation is an important area of research to improve the level of efficiency and autonomy in manufacturing processes. Due to the high precision and repeatability of industrial robot arms, robotic manufacturing tasks are dominated by simple pick, place, and peg insertion actions performed in a highly structured environment. Lego blocks are an excellent analog for studying robotic assembly as they are well structured in a grid along with only needing simple rotations and peg insertions to manipulate. They are also cheap, easily segmented with cameras, and robust to wear and tear.

In this paper, we explore both efficiency and autonomy improvements for Lego building tasks. We introduce "BrickPick," an actuated end effector capable of carrying multiple Lego blocks at once. The BrickPick end effector improves assembly efficiency by increasing the throughput of manipulated blocks per motion. We also introduce the "Vibro-Tactile Toolbox," a ROS software package used to integrate vibro-tactile feedback into manipulation skills. The toolbox provides a framework for utilizing sensor data from contact microphones, force-torque sensors, and cameras to inform the robot when actions should be terminated and what their outcomes have been. The Vibro-Tactile Toolbox improves assembly autonomy by providing an online self-supervisor to handle failed actions and facilitate the collection of labelled trial data.

Acknowledgments

Thank you to all who made this work possible!

To Zeynep Temel, my advisor. Your guidance and commitment to freedom helped me find new passions and interests through my research.

To Sarvesh Patil, my lab-mate. Your welcoming attitude and excitement helped me find my footing in the world of robotics research.

To ZoomLab. You are all so kind and willing to help one another.

To Kevin Zhang. It was a pleasure to work alongside you, I've learned so much in such a short time.

To Oliver Kroemer, Zackory Erickson and my collegues at Mil-19 for your additional support.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

Robotic manipulation is becoming increasingly more important in modern manufacturing processes, driving efficiency and precision in assembly lines. Industrial robots have revolutionized how products are made, allowing for tasks that require high levels of accuracy and repeatability to be automated. Among these tasks, simple actions such as pick, place, and peg insertion are fundamental, yet they hold significant potential for innovation. This thesis explores how advancements in robotic manipulation can further enhance the efficiency and autonomy of these essential manufacturing processes using Lego blocks as a system analog.

1.1 Motivation

The motivation behind this research stems from the growing demand for higher productivity and lower operational costs in manufacturing industries. As the complexity and variety of products increase, there is a pressing need for more flexible and intelligent robotic systems. Traditional industrial robots excel in structured environments with repetitive tasks but often struggle with adaptability and decision-making in dynamic settings. By improving the efficiency and autonomy of robotic manipulation, manufacturers can achieve greater throughput and reduce downtime, ultimately leading to cost savings and increased competitiveness. Additionally, the use of Lego blocks as a testbed provides a controlled, cost-effective, and scalable platform for developing and evaluating novel robotic techniques.

1.2 Problem Definition

Despite the advancements in robotic manipulation, several challenges remain for maximizing efficiency and autonomy in manufacturing processes. Current robotic systems are limited by their inability to carry multiple components simultaneously, which reduces throughput. Moreover, the lack of real-time feedback and adaptive decision-making capabilities hinders the robots' ability to recover from errors and perform complex tasks autonomously. This thesis addresses these challenges by introducing two key innovations: the "BrickPick" end effector, which enhances efficiency by enabling the robot to carry multiple Lego blocks at once, and the "Vibro-Tactile Toolbox," a ROS-based software package that integrates vibro-tactile feedback for improved manipulation skills. Operability is also improved by adding additional degrees of freedom to support objects in the workspace during manipulation events. These innovations aim to provide a more robust, efficient, and autonomous solution for robotic assembly tasks, paving the way for future advancements in industrial automation.

1.3 Foundational Works

In this section, we discuss the research and developments from CMU that have influenced our work with Lego block assembly in a major capacity. First, we will look at the development of the Robot Training Center (RTC) which emulates a manufacturing space with multiple arms, mobile robots, workspace cameras, and Lego pallets and supply depots. We will then look at the work of Intelligent Control Lab, who developed an passive Lego block manipulator that served as a precursor to the BrickPick end effector. Next, we will examine the integration of audio and force feedback data to improve the precision of robotic food cutting. Lastly, we will review the use of arm-mounted contact microphones in agricultural settings, which have been used to classify different types of objects (such as leaves, sticks, and branches) that collide with the robotic arm. These studies provide important insights and methodologies that have informed the developments presented in this thesis.

1.3.1 Robot Training Center

The Robot Training Center is a robotic manufacturing analog built at CMU's Mil-19 facility with the aim to study Multi-Level Planning, Execution and Control for Robotic Assembly and Disassembly. The stated goal of the project is: "Researchers will develop the component techniques to implement an integrated, multi-level system for task allocation and autonomous mobile robot (AMR) planning and control of a robotic Lego assembly/disassembly test bed at Mill 19 to be used to demonstrate advanced manufacturing capabilities " [10]. Our work is focused on the assembly side of the project to produce training data for improved manipulation policies which can be stored to the digital backbone. Figure 1.1 outlines the hardware setup for this task.

1.3.2 Robotic Lego Assembly and Disassembly

The earliest works in robotic Lego building at the RTC were published by the Intelligent Control Lab at CMU in "Robotic LEGO Assembly and Disassembly from Human Demonstration" [7]. This paper introduces a framework for robotic Lego assembly and disassembly, focusing on a novel end-effector tool (EOT), pick-and-place strategies, and build plans learned from human demonstrators. The EOT, inspired by the Lego separator tool, is a 3D-printed device designed for easy manipulation of Lego bricks through attachment and twisting motions. The system learns assembly and disassembly sequences from human demonstrations, using a depth camera and a pre-trained model to generate task information, verified by a digital twin for feasibility.

Deployed on a FANUC LR-mate 200id/7L industrial robot arm, the framework demonstrates its capability to learn and execute simple assembly and disassembly tasks effectively. The system addresses several critical constraints in Lego manipulation, such as alignment, geometric feasibility, and operability. Some limitations were discovered, including challenges in maintaining the operability of complex structures and handling the varying tightness of Lego connections. Despite these challenges, the study provides a robust and adaptable solution for automatic prototyping using Lego, setting a foundation for assembly and disassembly processes.

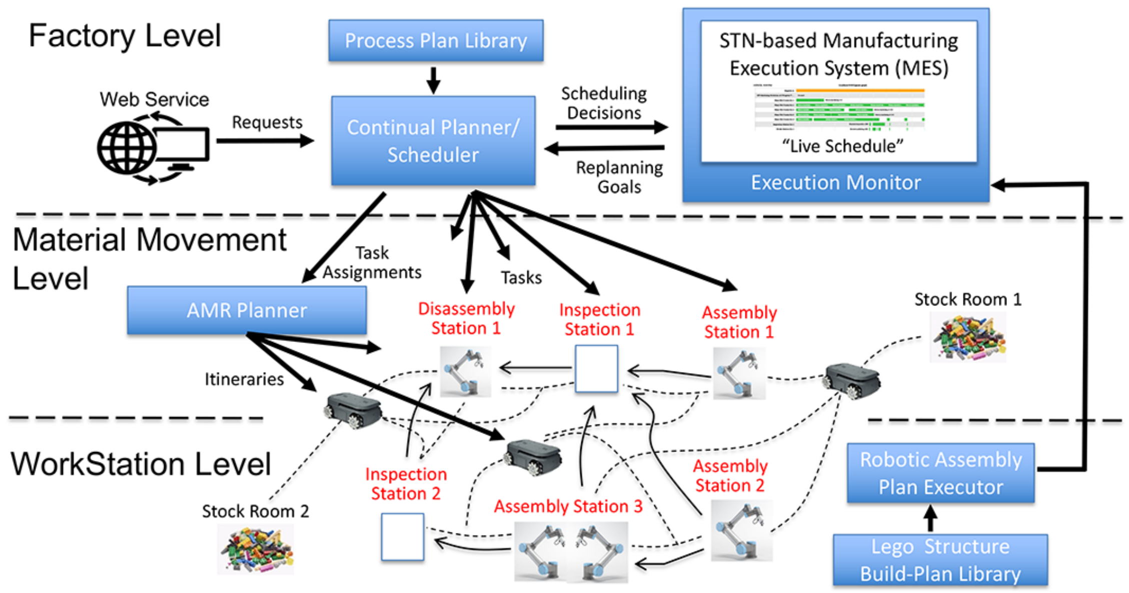


Figure 1.1: Robot Training Center: System Overview. Adapted from [10]

1.3.3 Leveraging Multi-modal Haptic Data

The foundational work for fused vibro-tactile feedback in robotic manipulation is established in the paper "Leveraging Multimodal Haptic Sensory Data for Robust Cutting" [12]. In this paper, a pair of Franka Robot Arms are fitted with a knife and pair of tongs with contact microphones positioned at the end of each arm and the cutting board. The robot uses haptic feedback from the joint control inputs for steady state tactile sensing and the contact microphones for dynamic tactile sensing through vibration. Fused together, the vibro-tactile data is used to train a cutting policy studied across a wide range of foods varying stiffness, friction, and other material properties. The vibro-tactile policy was shown to generalize to foods not used in training, demonstrating that the addition of audio/vibration data in haptic feedback loops can improve performance and robustness in dynamic manipulation tasks. The robot was also able to correctly infer based on vibro-tactile information when to terminate its cutting action when the policy detected the knife contacting the cutting board, laying the groundwork for vibro-tactile based action termination.

1.3.4 Audio Contact Classification

Audio-based contact classification has been studied in an agricultural context with the aim to identify when a robot arm has collided with leaves, sticks, branches, or other cluttering objects. In the paper "Contact Classification for Agriculture Manipulation in Cluttered Canopies" [5], piezo-electric contact microphones are fitted to the linkages of a robot arm to detect vibrations from external contacts. The audio signals are extracted to Mel Frequency Cepstrurm Coefficients (MFCC Features) to provide a more distinguishable signal profile for distinct contact modes. The MFCC representations of audio are then used to train a deep neural network to classify each contact as leaf, stick, or branch, which may inform a robot planning or control algorithm when to stop an action on account of excessive canopy clutter. This work demonstrates that short horizon contact events can be distinguished using audio signals, establishing a successful example of audio based contact classification.

1.4 Literature Review

Manipulation is an important area of research for robotics, as it is difficult to accomplish without costly investments to environment structure. In order to reduce the degree of structure required to successfully execute a manipulation task, a number of established strategies can be used. Cameras are an excellent tool to identify objects, targets, and obstacles within a robot's workspace in order to localize items of interest in real time [6]. Utilizing RGB-D cameras can further improve the localization accuracy by providing a full 3D colored point cloud of any viewed object, allowing a system to construct a digital twin of the physical space. Localizing objects with cameras can produce a more robust manipulation system with a reduced reliance on known pose targets [11]. Wrist-mounted cameras can also be used for visual servoing tasks, allowing a robot to engage an object with precision without necessarily knowing where the object is located in its workspace [3].

Historically, industrial robots with a high gear ratio at each joint have been used in robotic manufacturing for their high degree of repeatability, precision, and resistance to loading [4]. These robots are excellent at driving their end effectors to target positions, but often struggle to perform dynamic tasks or respond to unexpected stimuli. Other studies propose the use of quasi-direct drive robot arms for force-controlled manipulation tasks in order to increase robot dexterity and operability in dynamic environments [1].

Outcome detection through both visual and tactile sensing can also be used to inform a manipulation system of how to proceed with execution. Learning complex motor skills for real world manipulation tasks is challenging, as it can be difficult to provide ground truth labels or rewards for learning models. Outcome detectors can be used to fine tune a skill beginning with a coarse initial demonstration, allowing a robot to explore its action space during training [9].

1.5 Contributions

Three major contributions are identified in this paper: The BrickPick end effector, the Vibro-Tactile Toolbox, and the integration of these items with the Robot Training Center. The development of the BrickPick end effector provides a means to increase assembly throughput by enabling a robot arm to manipulate multiple blocks in a single payload all at once. The Vibro-Tactile Toolbox provides the necessary software tools to utilize audio, haptic, and visual feedback in the supervised execution of manipulation skills. The toolbox also contains resources for data collection and training vibro-tactile based learning models to deploy in future iterations of the project. Last, we include deployment tools and API templates to allow the vibro-tactile toolbox to be repurposed for robotic manufacturing uses outside of Lego block assembly.

Chapter 2

BrickPick End Effector

We begin by exploring the development of an end effector specialized for the task of picking and placing Lego blocks. To simplify the development process, the first major goal for the end effector was to pick and place basic Lego blocks onto a fixed build plate. "Basic Lego blocks" entails blocks of nominal sizing equal to or less than 4 studs in width. The first approach was to retro-fit the Lego brick separator onto a robot arm, as there already existed a part specifically designed for removing Lego blocks from studs. The tool has three main features which can be utilized for pick and place tasks: Studs to firmly connect the tool to the target brick, A corner to allow the Lego to be rotated by the tool, and a lever arm to increase the applied prying torque. These features work together to allow a user to easily connect to and pry off a Lego from the structure below through an applied and amplified torque. This approach was very effective for picking and placing basic Lego block sizes (2x1, 2x4, 4x1, 2x2), establishing the connect and rotate strategy as the foundation for future iterations in end-effector design.

An end-effector was developed by the Intelligent Autonomous Manipulation Lab which features similar geometry to the Lego separator tool, but with the the lever arm removed. This is done to reduce the footprint of the end effector, as a robot arm is generally capable of applying sufficient torque for removal without the need for an additional lever. The part is composed of a 3D printed extrusion with a slot to glue a 2x1 Lego brick to the bottom. For improved reliability, the corner features chamfer to help feed misaligned bricks into the glued Lego brick. For picking actions,

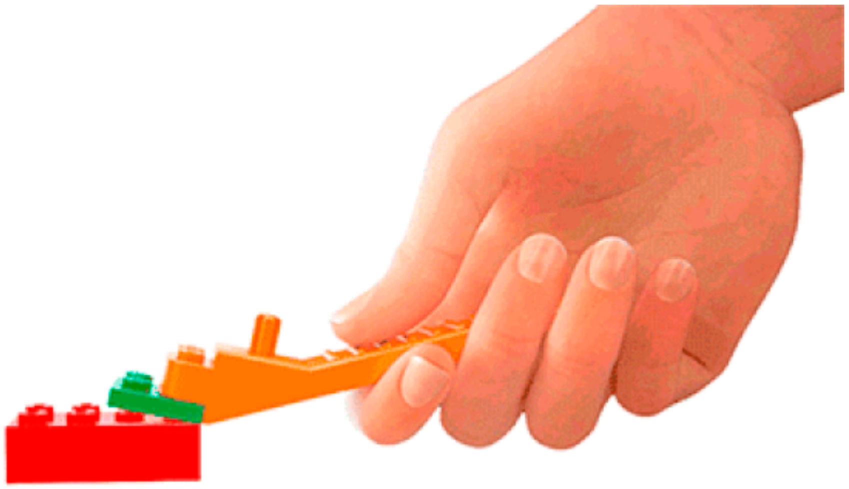
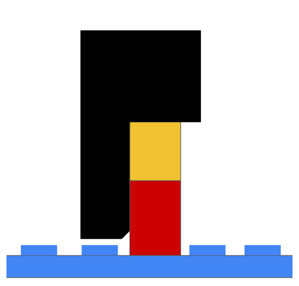
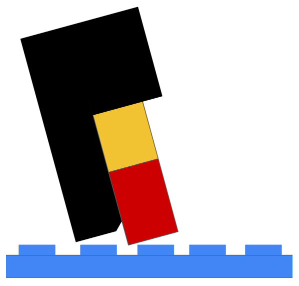


Figure 2.1: Lego Separator Tool. Adapted from [2]

the target brick is rotated about its bottom edge towards the corner or ”moment plate,” maintaining the connection to the end effector through rotation. For picking actions, the end effector is rotated about the top edge of the target brick away from the moment plate to release during rotation.

We first discuss the development of the BrickPick end-effector, which aims to increase the capability of the passive gripper by adding extendable moment plates to facilitate the picking of multiple Lego blocks in a stack and a plunger to deposit the full payload. For all revisions, the pick strategy remains the same as the passive gripper. Revision 1, named ”BrickPick,” is a proof of concept design with the goal of showing that extendable moment plates can be used to increase the payload capacity of the end effector. This proof of concept design is made to fit to a large UR5 robot arm and tested on a single stack of 2x4 bricks. Revision 2, named ”BrickPick-Compact,” is closer to a Minimum Viable Product (MVP) which is suited to complete full assembly tasks mounted to a smaller Yaskawa GP-4 robot arm in the Robot Training Center. This revision seeks to reduce the size of the end effector by replacing the


(a) Gripper brick (yellow) is connected to target brick (red).


(b) Target brick is picked from build plate by rotating gripper about its bottom left edge.

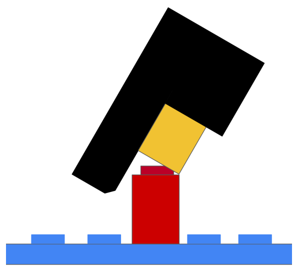

(c) Target brick is placed on build plate by rotating gripper about its top right edge.

Figure 2.2: Passive Gripper - Pick and Place Motions

large linear actuators as well as reducing cord count with a wireless interface and battery. Revision 3, named "BrickPick-Tactile," is the final polished prototype of the BrickPick end effector. Soldered prototype electronics are replaced with a PCB and overall part count is reduced. This revision is far more reliable and further reduces the footprint of the end effector, limiting the workspace constraints that must be applied to accomplish an assembly task.

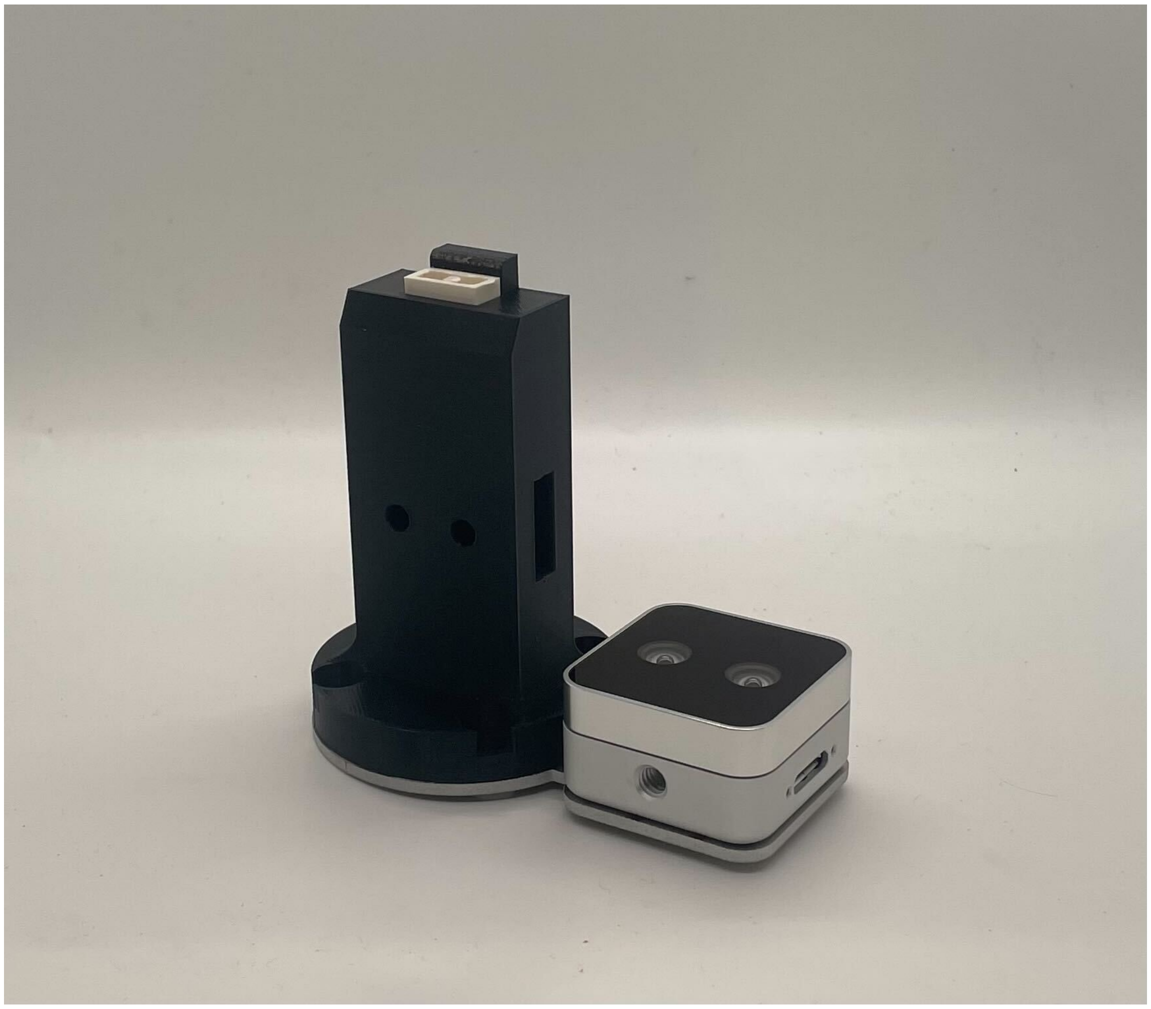


Figure 2.3: Passive Gripper: Isometric View

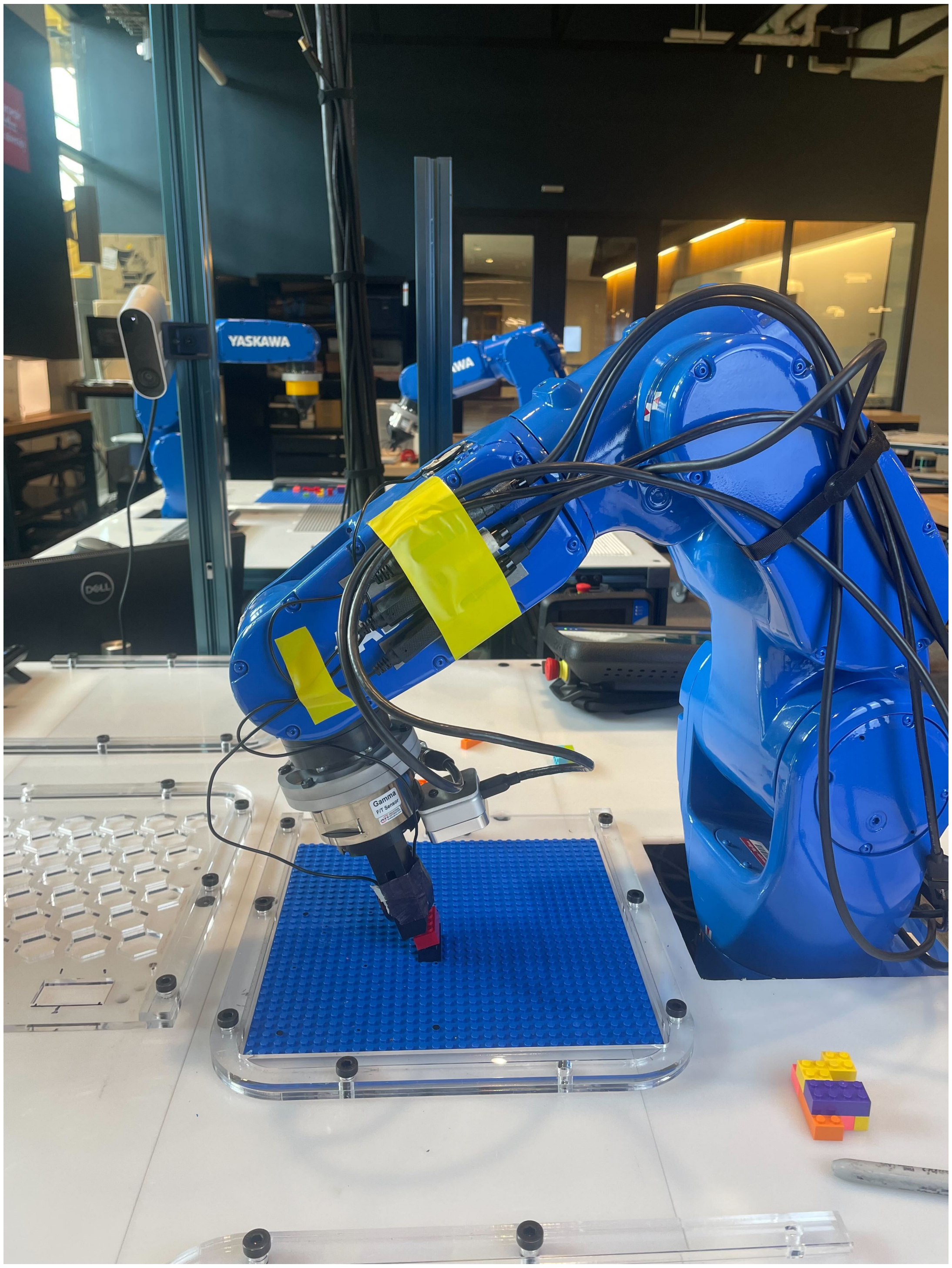


Figure 2.4: Passive Gripper: Picking a brick with a Yaskawa-GP4 robot with a stack size of 2.

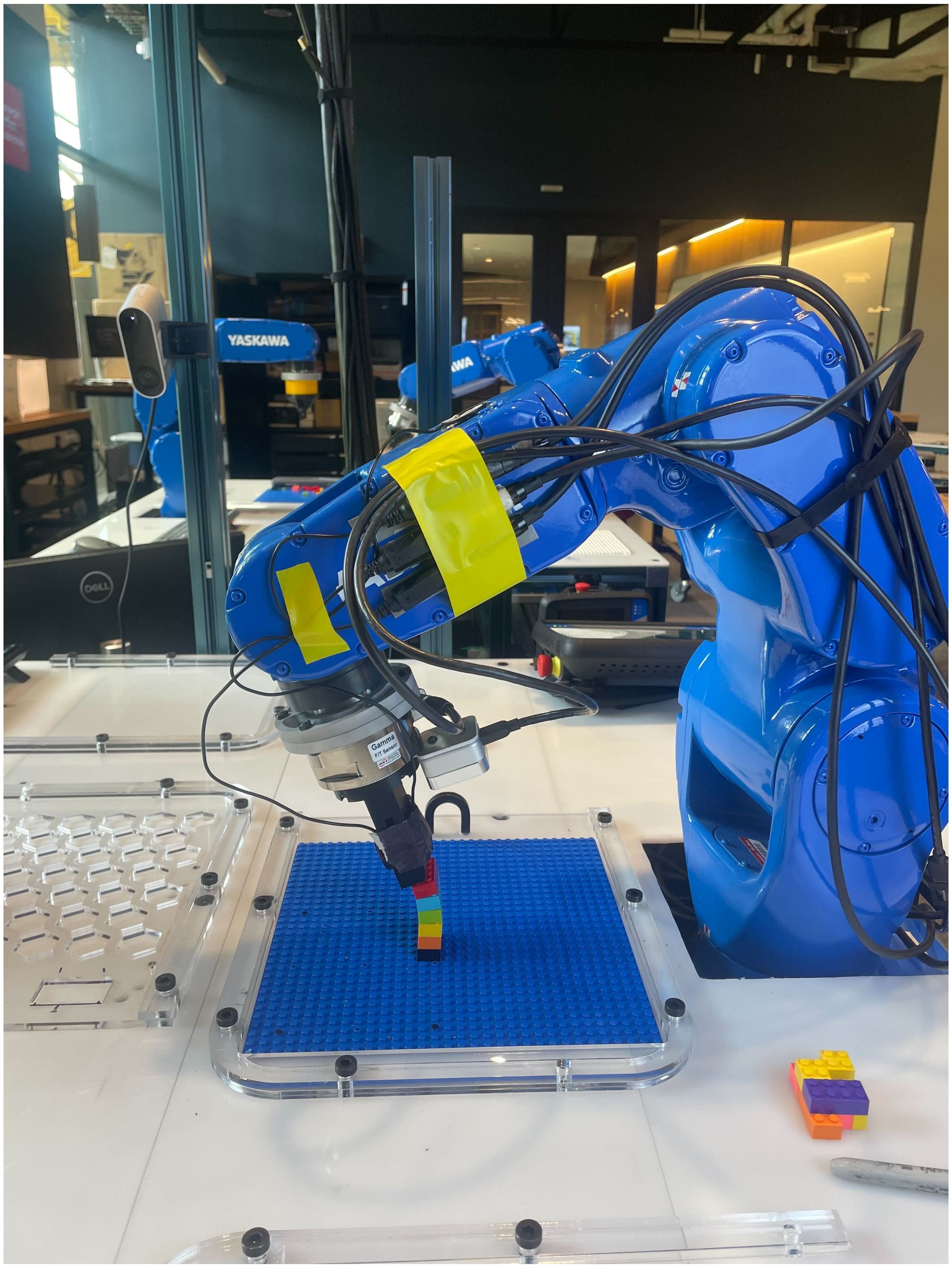


Figure 2.5: Passive Gripper: Picking a brick with YK-Creator with a stack size of 6. Block stack is too tall to support the applied picking torque.

2.1 Revision 1: BrickPick

The first revision of the BrickPick end effector was developed to test the function of extendable moment plates and an actuated plunger for Lego manipulation. Basic Lego blocks are 9.6mm in height so linear actuators with a stroke length of 50mm are selected to accommodate a payload size of five Lego blocks. A plate is added to both the long edge and short edge of the base block in the event that a target brick is wedged between two others on the build plate. The moment plates are used to support the payload through rotation, allowing the end effector to control the loading experienced by the Legos left on the build plate. This is an important design feature, as the passive gripper was unable to prevent picking torques to transmit through a stack of blocks, limiting its ability to pick up portions of a stack. To accommodate the new space requirements of linear actuators, the 2x1 base brick is replaced with a larger 2x4 brick with holes for a plunger. Last, a linear actuator is added with plunger rods to deposit the payload without rotation.

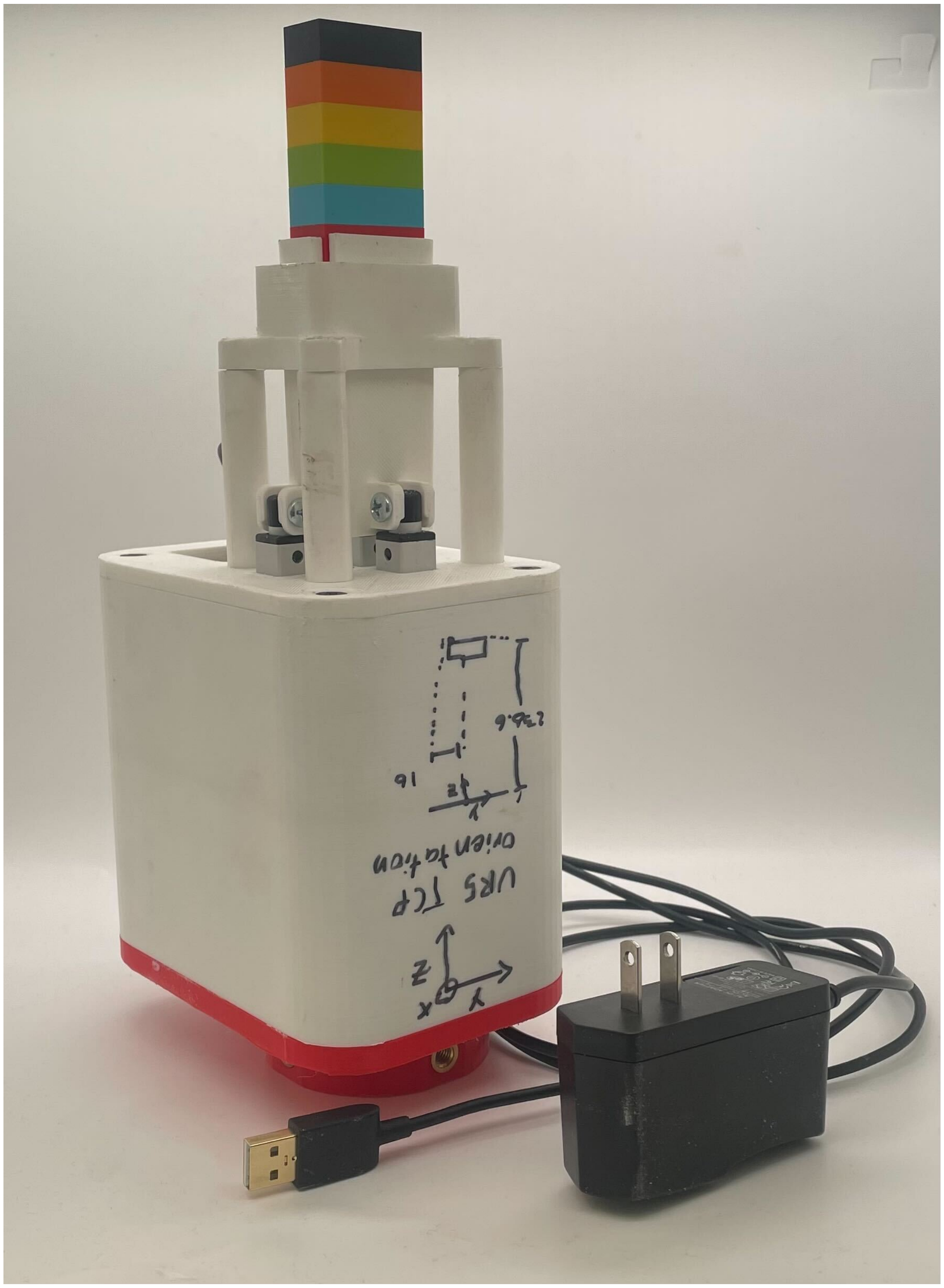


Figure 2.6: BrickPick: Isometric View

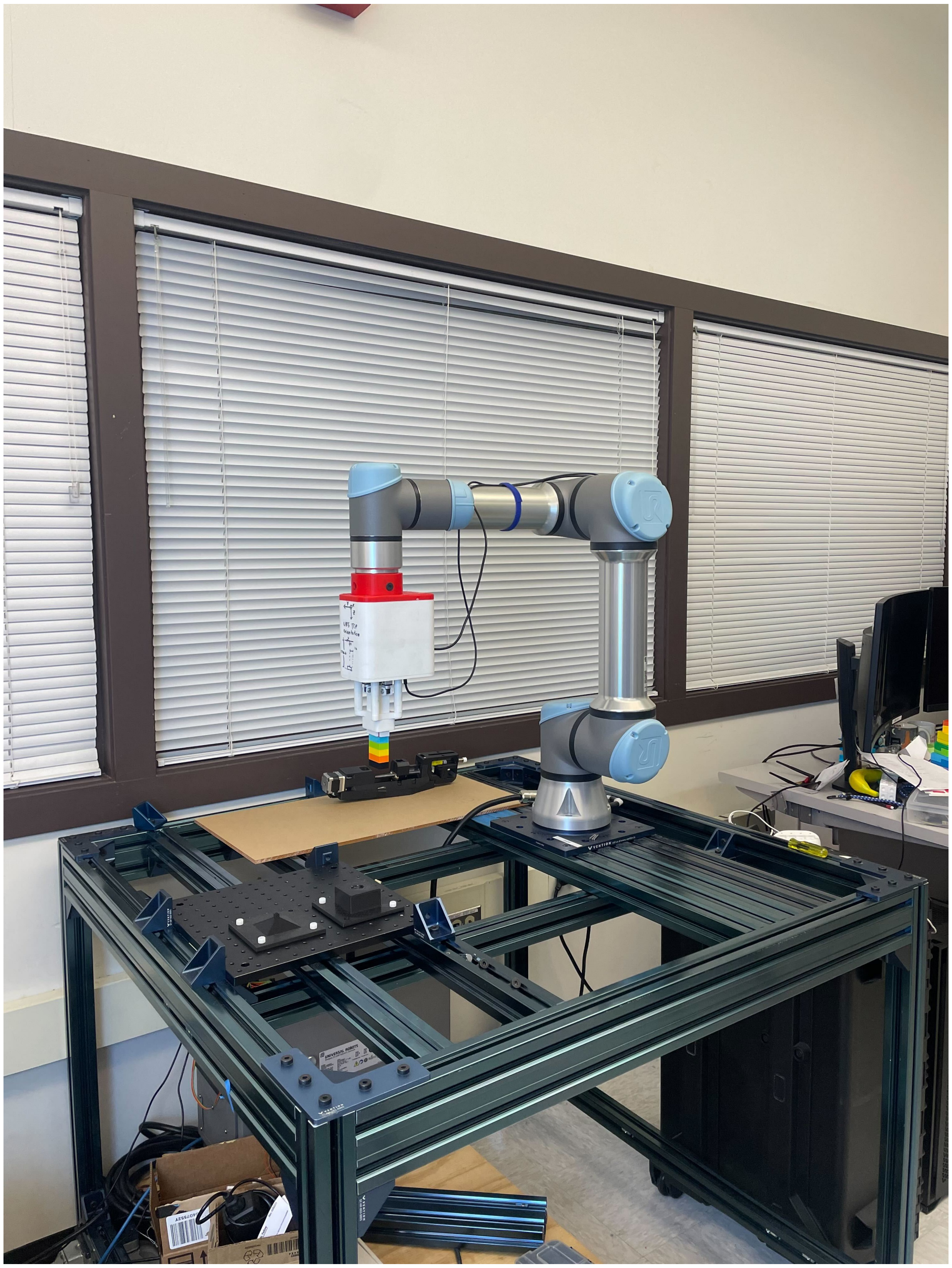


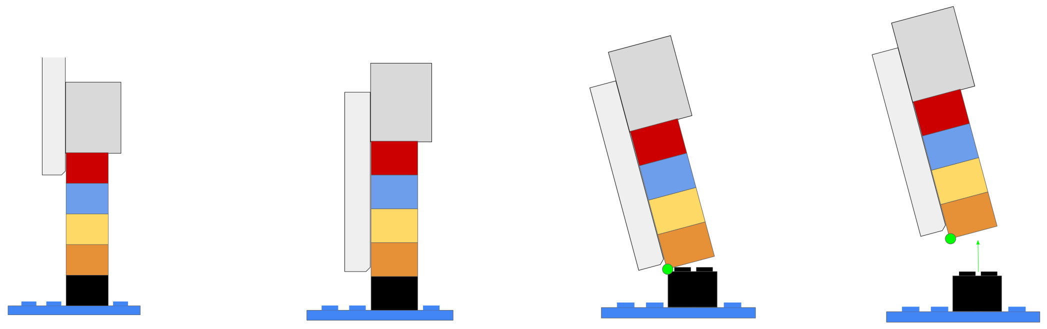
Figure 2.7: BrickPick: Mounted on UR5e Robot Arm

2.1.1 Pick and Place Strategy

The picking motion employed by the BrickPick end effector is similar to that of the passive gripper. First, the end effector brick is connected to the target brick 2.8a. Second, a moment plate is extended above the target brick 2.8b. Although extending the plate further increases grip, a small clearance is necessary to prevent contact with the 1.7mm tall Lego studs when rotated up to 30 degrees. Third, the target brick is rotated about its bottom left edge 2.8c. The clamp force between end effector and target brick is now larger than that between the target brick and build plate. Last, the gripper is pulled up to release the target brick from the build plate 2.8d.

During rotation, torque from the end effector causes the payload stack to bow or buckle, especially with taller stacks. The moment plate reinforces the stack, ensuring a stable connection between the end effector and the Lego blocks. However, placing tasks pose challenges, as the moment plate must retract to deposit the payload. Without the plate's support, stacks taller than two bricks tend to bend or fall apart during rotation.

To address this, payload placement is achieved using a plunger mechanism. A linear actuator with a 10mm stroke length pushes three plunger rods into the top of the stack, separating it from the end effector brick. This strategy avoids buckling by applying a simple force parallel to the payload studs, rather than relying on torque and rotation when a support plate cannot be used.



(a) Gripper Connected (b) Plate Extended (c) Brick Rotated (d) Brick Released

Figure 2.8: BrickPick - Picking Process

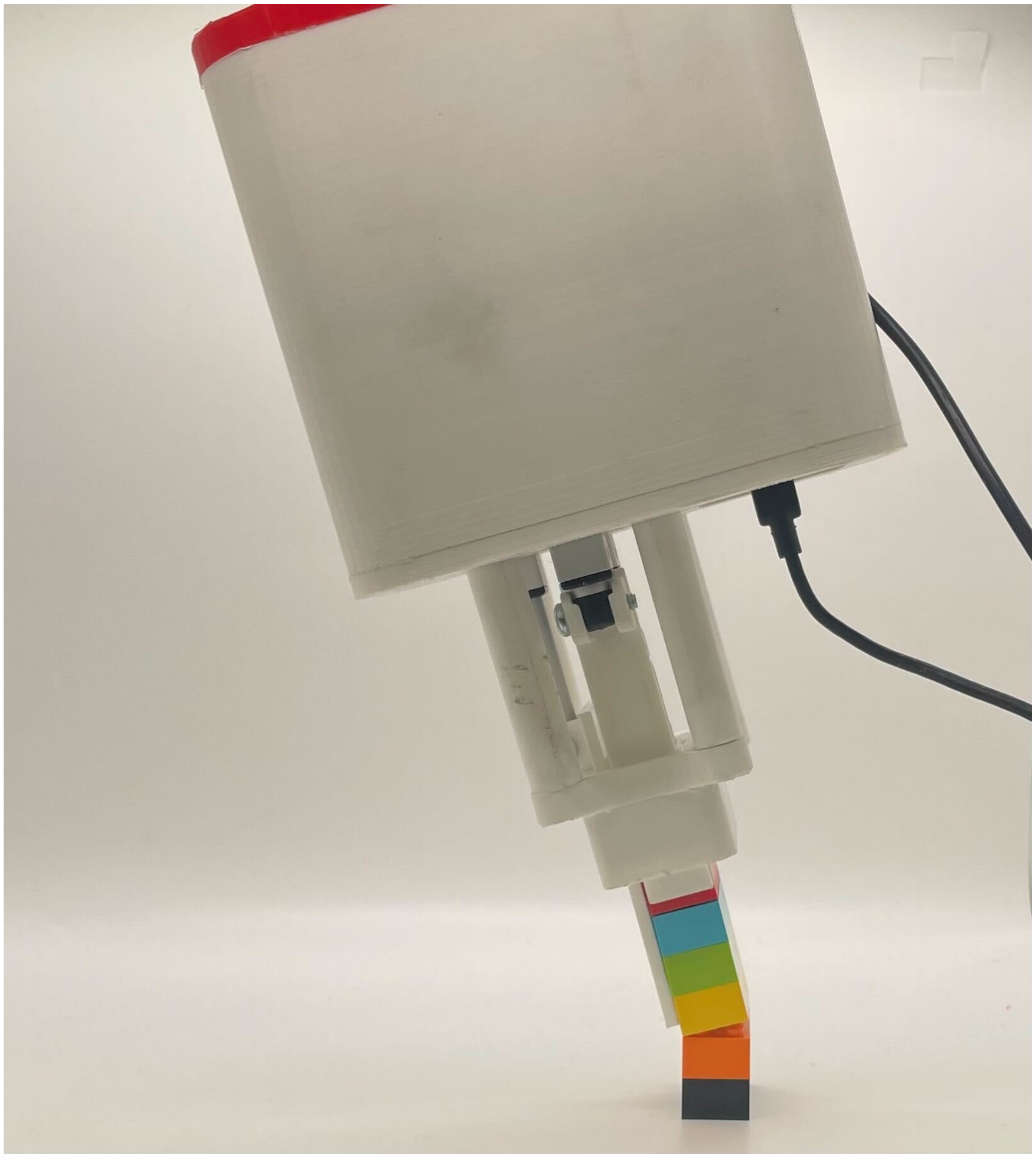


Figure 2.9: BrickPick: Picking 3 blocks with the long moment plate.

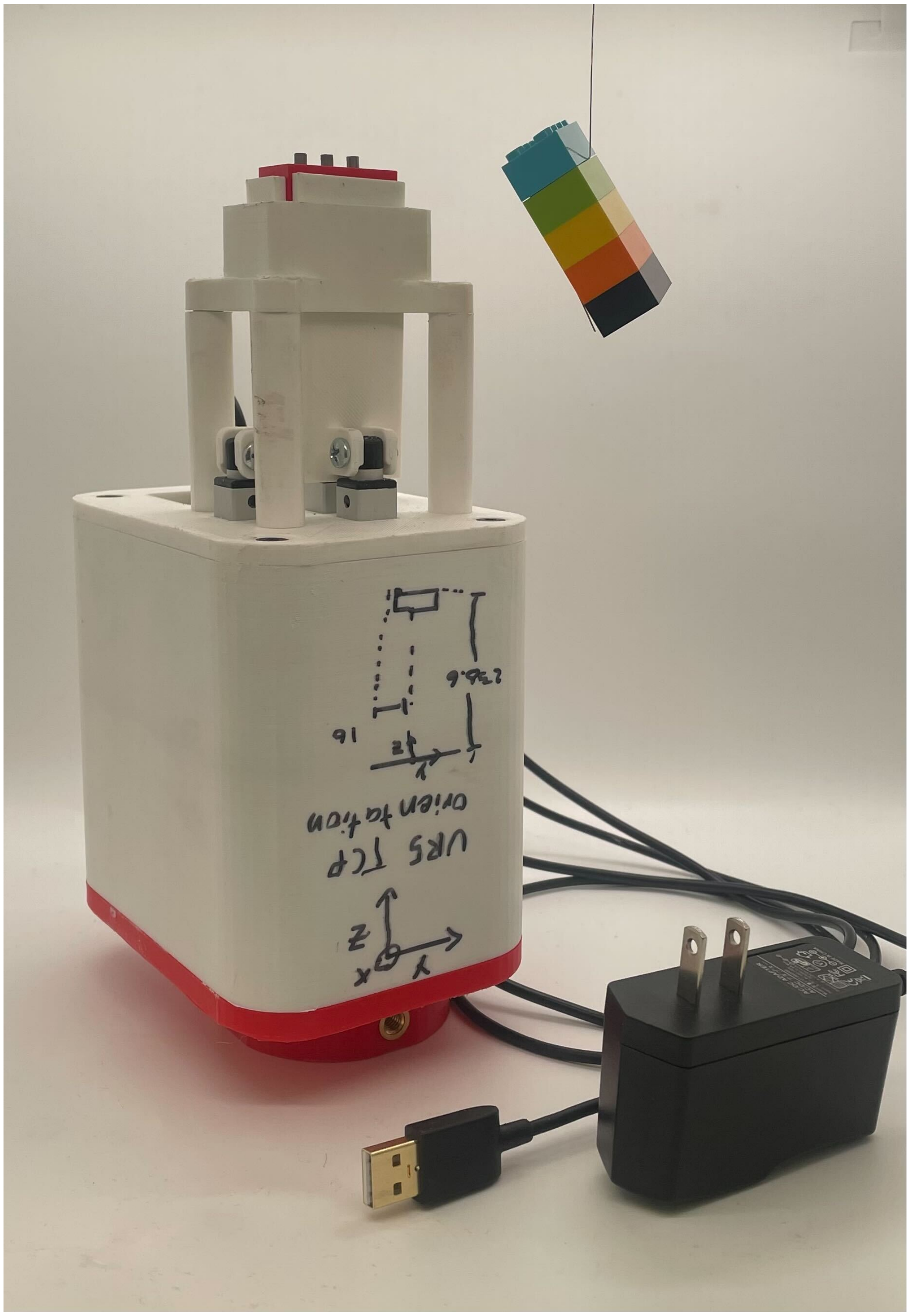


Figure 2.10: BrickPick: Depositing payload with plunger

2.1.2 Actuator Stack

For picking and placing actions, the BrickPick end effector uses linear motion, with the robot arm providing rotations after the moment plates are set. Linear actuators with potentiometer feedback allow for simple feedback control of the moment plates and plunger extensions. Each moment plate uses a linear actuator with a 50mm stroke length, enabling traversal of up to 5 Lego blocks. However, due to actuator housing, extension, and moment plate lengths needing to match the stroke length, the maximum extension of the plates in BrickPick revision 1 is limited to 35mm, accommodating 3 Lego bricks. The plunger requires a stroke length of only 1.7mm (the height of a Lego stud), so a 10mm stroke actuator was selected.

The moment plates are attached in series to the linear actuators, with the actuator housing fixed to the end effector base and the plates extending past the moving shaft along the payload length. This design is space-efficient in width but requires the end effector to be at least twice the length of the largest payload. Later revisions address this length inefficiency for smaller robots and crowded workspaces. The plunger actuator, connected in series with the plunger body and rods, does not affect the overall footprint, as it is much shorter than the plate stack.

2.1.3 Electronics

We start by selecting Actuonix L12-XX-P series linear actuators for their compact design with a stroke length of 50mm for the moment plates and 10mm for the plunger. The linear actuators operate at 6V with a 3-pin potentiometer to measure the displacement of the moving shaft. An Adafruit Feather M0 is selected as the micro-controller for the BrickPick end effector for its small size and modularity. It is stacked onto an Adafruit Featherwing Motor Shield to power/control the 6V linear actuators with a barrel jack. An Adafruit ADS 1015 Analog to Digital Converter (ADC) is used to convert the voltage of the potentiometers into a readable digital signal. The micro-controller features a USB-A socket which is used for wired serial control.

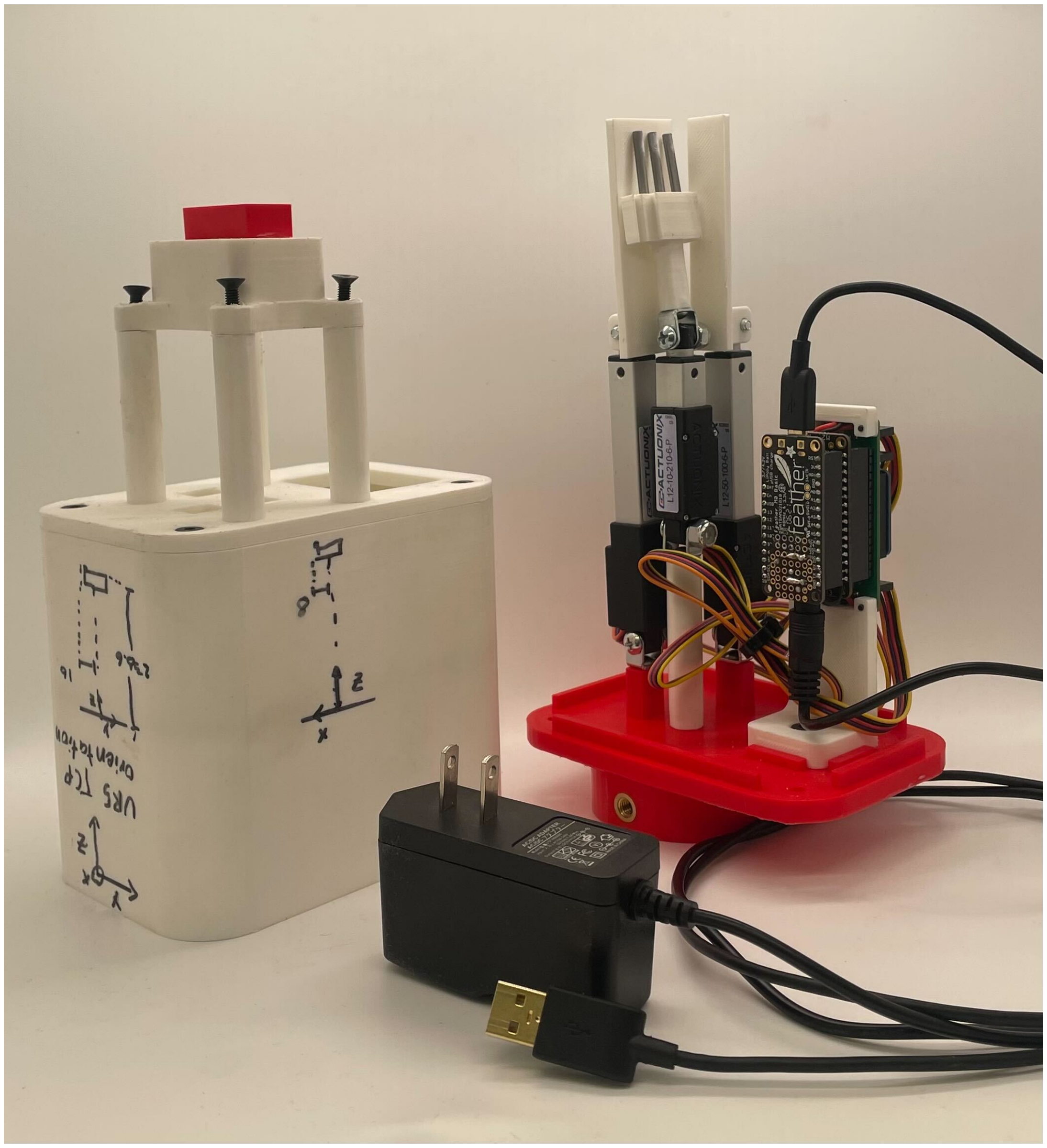


Figure 2.11: BrickPick: X-Ray view.

2.1.4 Embedded Code

The embedded code deployed to the BrickPick micro-controller contains two major modules: A PID controller to drive the linear actuators, and a serial control interface designed for manual human input. The linear actuators are slow and powerful, but contain a large control deadzone ending near 10% of the total control input. Hysteresis effects can cause PID controllers to exhibit thrashing near their target position as the friction causes the actuator to overshoot and build integral error in a repeated process. To overcome this, the system dynamics were measured to build a correspondence between control input and velocity. The correspondence was used to create a feed-forward control policy using ramp trajectories to more accurately approach the target position. The feed forward control policy drastically improved the settling time of the system by reducing overshoot and changing the PID error term from distance to target to distance to expected trajectory position. Shown in 2.12 is a control curve of the target state vs measured state.

Command	**Description**	**Parameters**
'H'	Print help message	None
'G {q1} {q2} {q3}'	Goto position (mm)	q1: Long plate q2: Short plate q3: Plunger
'B {plate} {brick}'	Goto brick	plate: 0=Long,1=Short brick: Brick to go to
'P'	Toggle Plotting	None
'D'	Deposit payload (Extend plunger)	None
'R'	Retract plunger	None

Table 2.1: BrickPick - Serial Command Interface

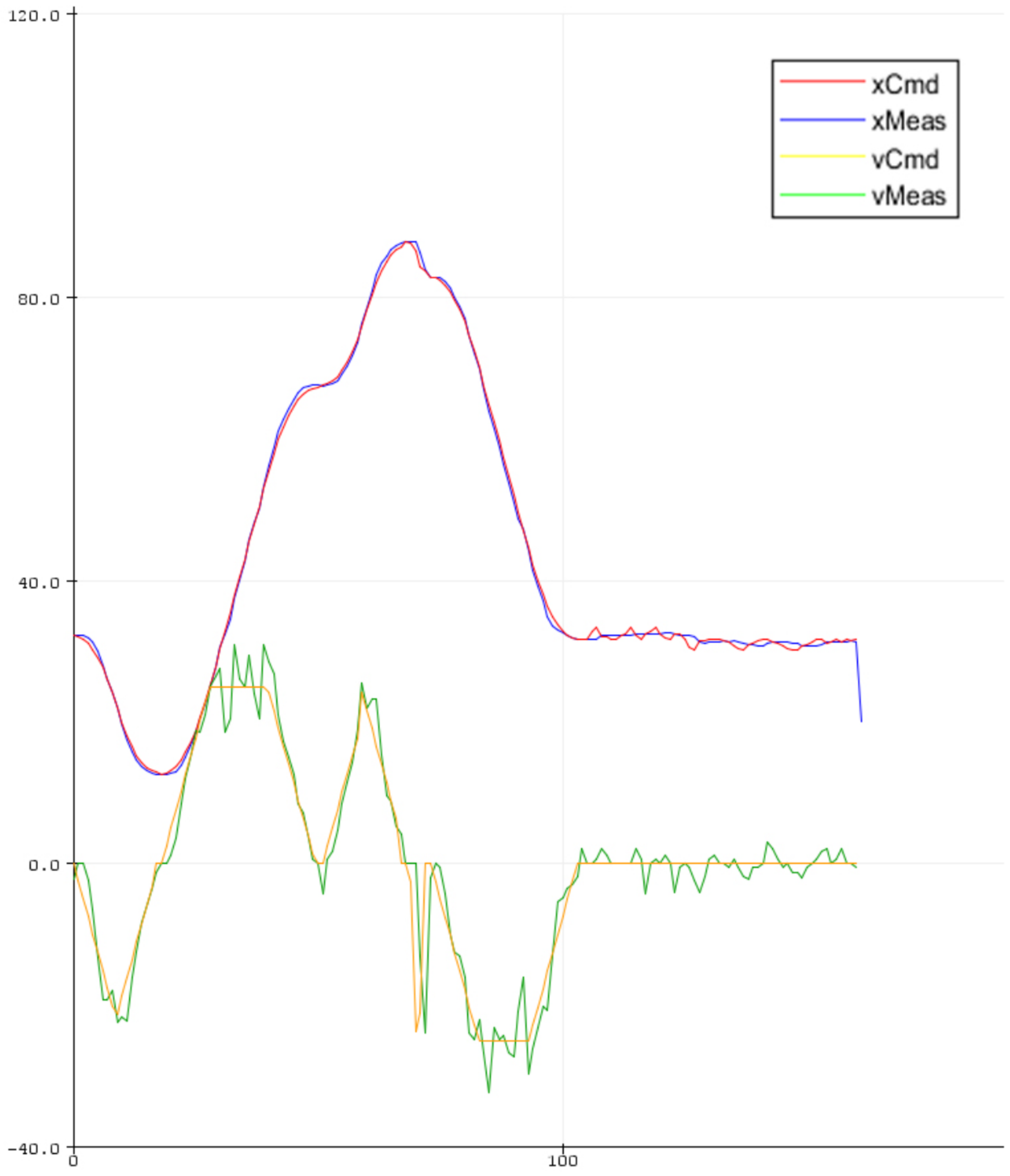


Figure 2.12: BrickPick: Control curve of a linear actuator with PID and FF ramp trajectory.

2.2 Revision 2: BrickPick-Compact

BrickPick-Compact is the title given to the second revision of the BrickPick end effector, developed to address the operability problems identified with the first revision. With the next goal of producing a minimum viable prototype to be deployed to the YK-Creator arm in the Robot Training Center, a few changes must be made. First, the robot must be made smaller in length in order to fit to the workspace of the smaller Yaskawa-GP4 robot arms. Second, the moment plates must be made longer and stiffer to accommodate a full sized 5 block payload. Third, the end effector must accommodate the use of an Intel Realsense D405 wrist-mounted camera. Last, reducing the cable count of the robot is an important upgrade as YK-Creator comes fitted with 2 wires for contact microphones, 1 wire for a wrist-mounted force torque sensor, and 1 wire for the wrist-mounted camera.

The length is reduced by replacing the actuator stack for the moment plates with a rack and screw design, moving material away from the length and out towards the width. The drive screws are placed near the tip of the end effector to allow the moment plate racks to extend out from within the end effector housing by up to 60mm, longer than the size of a full 5 brick payload. Parts of the housing are removed to give the wrist camera full view of the payload and build plate while mounted onto BrickPick-Compact. Space for a LiPo battery is added along with exchanging the microcontroller for an Adafruit Feather ESP32-S3 TFT board which is WiFi enabled. This reduces the external cable count of the end effector from 2 to 0 (not including the wrist camera).

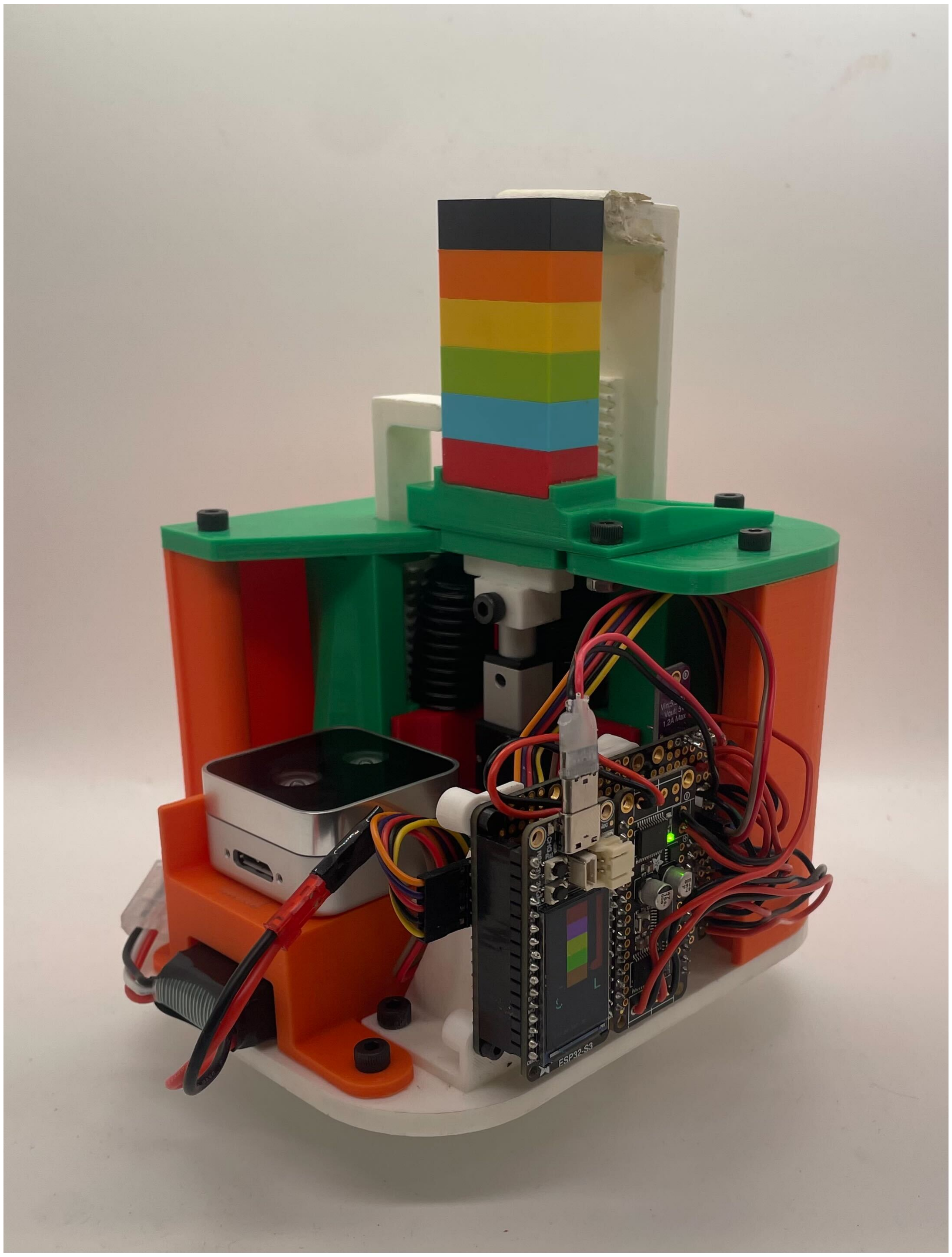


Figure 2.13: BrickPick-Compact: Isometric View

2.2.1 Actuator Stack

To reduce the length of BrickPick-Compact, linear actuators are replaced with a custom screw-rack mechanism which serves to flatten the moving shaft of the actuator into the moment plate body reducing overall length by approximately the size of a full 5 block payload stack (50mm). Fully contracted, the length of the BrickPick-Compact actuator stack is 118mm. This is a significant improvement from the serial design with linear actuators made for BrickPick which has a fully contracted length of 196mm. A cross section of the screw-rack mechanism is shown in 2.14

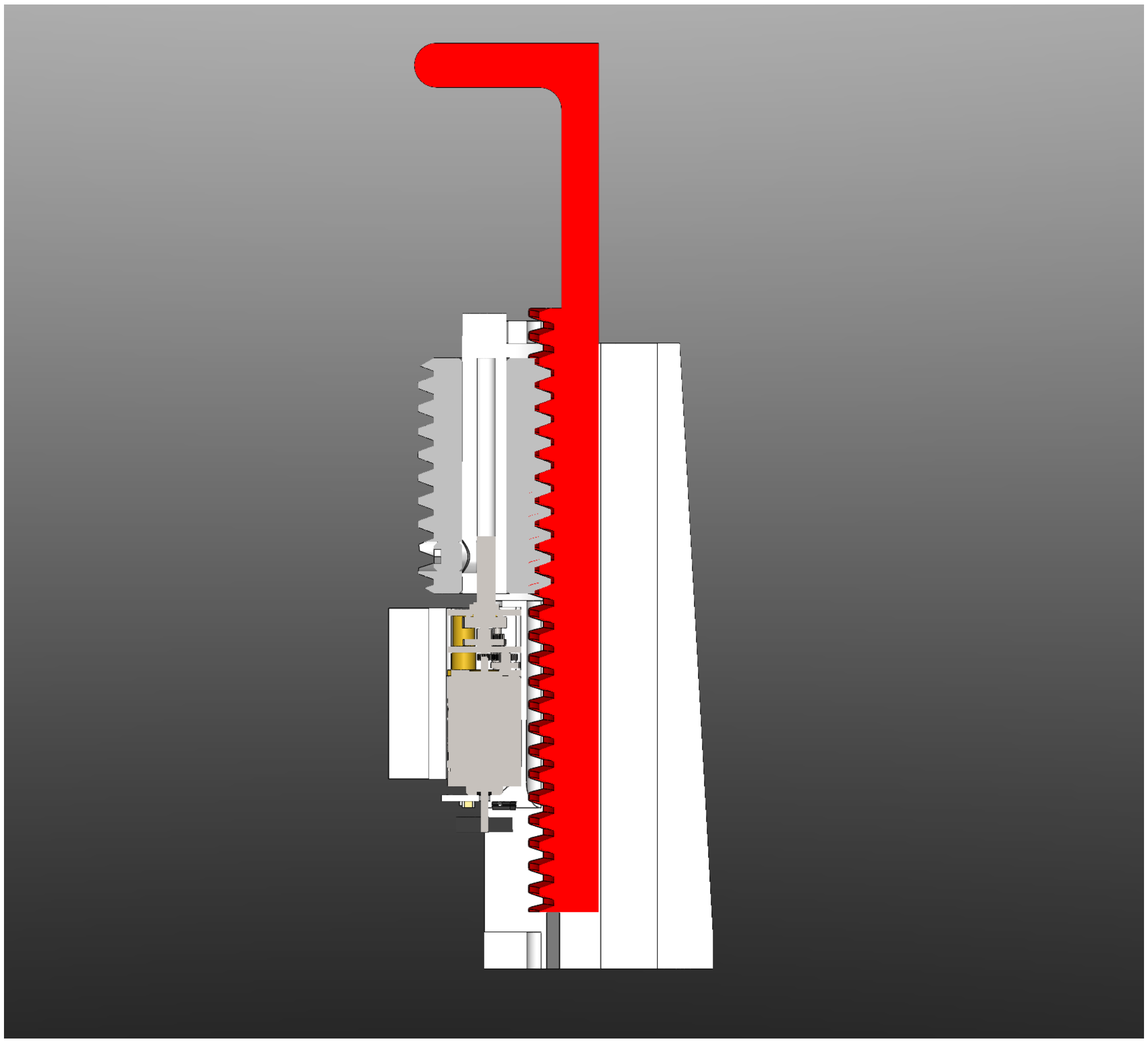


Figure 2.14: BrickPick-Compact: Screw-Rack actuator stack, cross-section view.

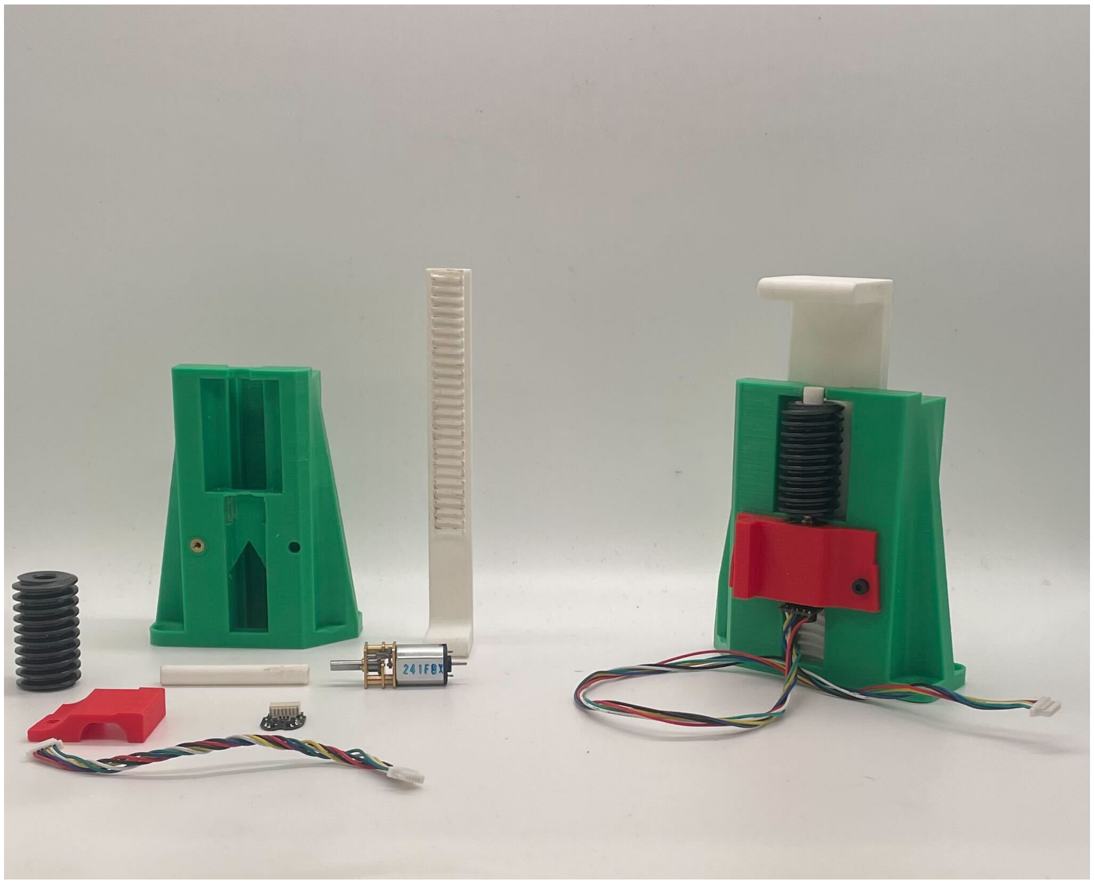
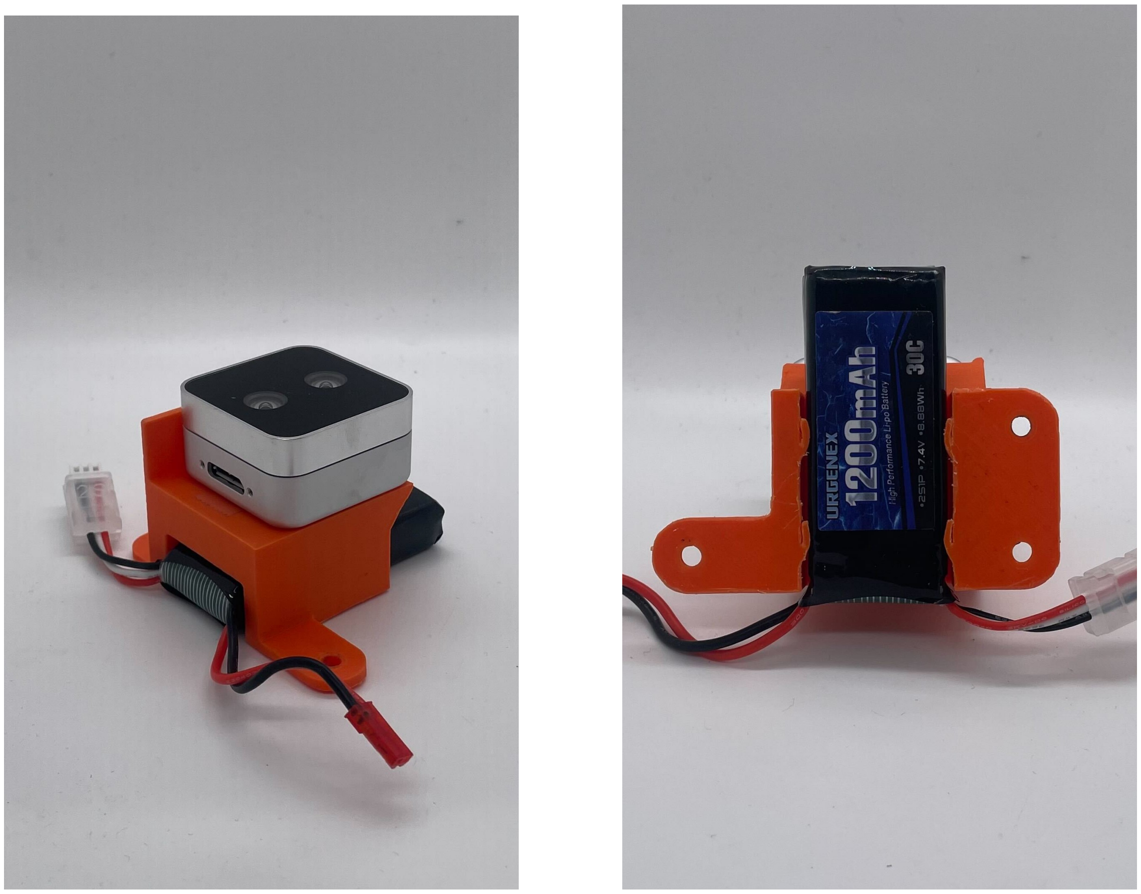


Figure 2.15: BrickPick-Compact: Actuator stack and component breakdown.

2.2.2 Electronics

The electronics of BrickPick-Compact are selected to remove external cable requirements to power and control the gripper. An Adafruit ESP32-S3 TFT is used as the microcontroller for its WiFi capability. Instead of serial control, BrickPick-Compact acts as a wireless server, accepting HTTP requests from clients across a WiFi network. The 6V power cable is replaced with a 7.4V LiPo battery which connects to a 6V and 3.3V buck converter to power the motors and microcontroller respectively. The external ADC is removed as the microcontroller is equipped with analog input pins. A mount for an Intel RealSense D405 stereo camera is added with compliant collet to holster the battery 2.16.



(a) Isometric View (b) Battery Collet

Figure 2.16: BrickPick-Compact: Camera-battery mount.

2.2.3 Embedded Modifications

BrickPick-Compact acts as a WiFi server and can be controlled using HTTP GET requests. The list of commands is extended by adding direct velocity control to each motor to aid in resetting the positions of the moment plates. BrickPick-Compact commands are now also made blocking or non-blocking to inform a larger robotic system of when a command has been completed successfully. The revised set of commands is shown in Table 2.2. A python adapter client was written to wrap commands input by the user with HTTP GET request bodies to pass the command with parameters to the BrickPick-Compact server. If a blocking command request is received, a response will not be sent to the client until the command has been completed. The response will include a flag to inform the client if the command terminated successfully or not.

Listing 2.1: BrickPick-Compact: HTTP client command request

```
GET http://{command}/?{param1}={param1_val}&... HTTP/1.1
Host: {ip_address}
User-Agent: python-requests/2.25.1
Accept-Encoding: gzip, deflate
Accept: */*
Connection: keep-alive
```

Command	**Description**	**Parameters**	Blocking?
help	Print help message	None	No
reset	Fully retract all actuators Deactivate PID	None	Yes
stop	Set all motor inputs to 0 Deactivate PID	None	No
set_short_ctrl	Set short plate control to u Deactivate short PID	$u \in [-255, 255]$	No
set_long_ctrl	Set long plate control to u Deactivate long PID	$u \in [-255, 255]$	No
set_short_target_brick	Set short plate target to b Activate short PID	$b \in [0, 5]$	Yes
set_long_target_brick	Set long plate target to b Activate long PID	$b \in [0, 5]$	Yes
deposit	Extrude plunger	None	Yes
raise	Retract plunger	None	Yes

Table 2.2: BrickPick-Compact: HTTP Command Interface

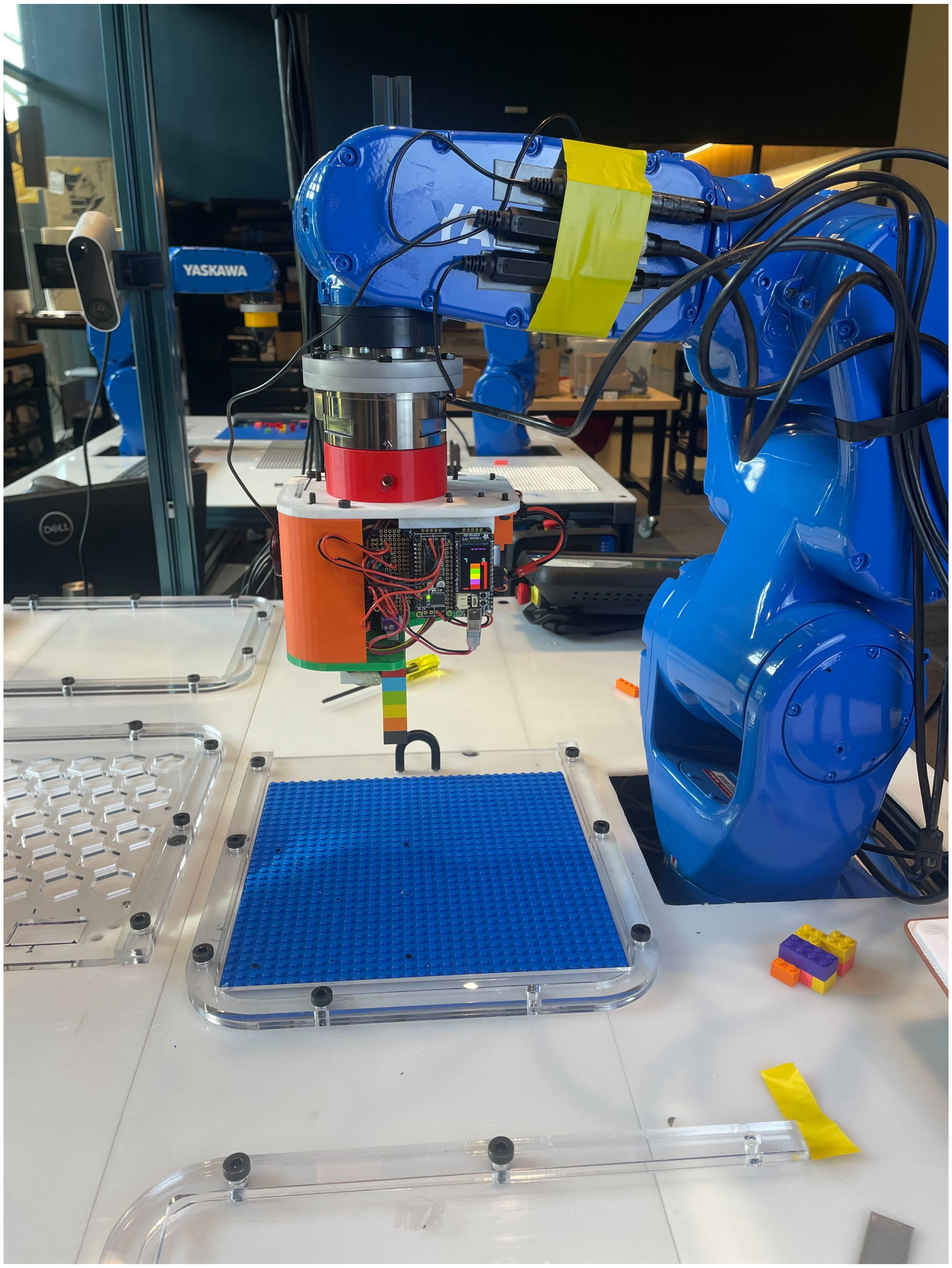


Figure 2.17: BrickPick-Compact: Mounted to YK-Creator.

2.3 Revision 3: BrickPick-Tactile

BrickPick-Tactile is the third and final revision of the BrickPick end effector built to be a polished prototype. The mechanical housings are reduced to a single main body improving the footprint and strength of the design. The fully retracted length of the actuator stack is further reduced to 109mm in length with angled plate tips to provide more clearance when rotating to pick a stack. The electronics are moved from a soldered protoboard to a fabricated PCB with sockets to solder the microcontroller, motor shield, and 6V buck converter. The TFT display included with the ESP32-S3 board contains a debugging display showing the IP address of the end effector, the PID targets, and the recorded position of each moment plate.

2.3.1 Mechanical

BrickPick-Compact was short enough to fit to the workspace of the Yaskawa robot arm, but its width near the gripper block prohibited its use in cluttered environments. To further relieve workspace constraints, the footprint of BrickPick-Tactile was made in a more conical shape by reducing its width near the tip. This allows the end effector to operate more freely near other stacks of bricks on the build plate. One of the ways this is accomplished is by moving the actuator screws to the exterior of the housing instead of interior. This allows the moment plates to move inwards towards the payload stack, reducing the width of the extended plates. The plate 'fingers' are also angled to allow for a larger range of rotation when picking bricks directly from the build plate.

Reducing the many housing components to a single body allows for a further reduction in size with the added benefit of improved stiffness and tolerance between major structures. Smaller clearances are used for the slots to constrain the sliding of the moment plates which reduces the amount of play in the structure and helps prevent binding. The moment plates are now fabricated using resin SLA printing instead of FDM for an improved rack quality.

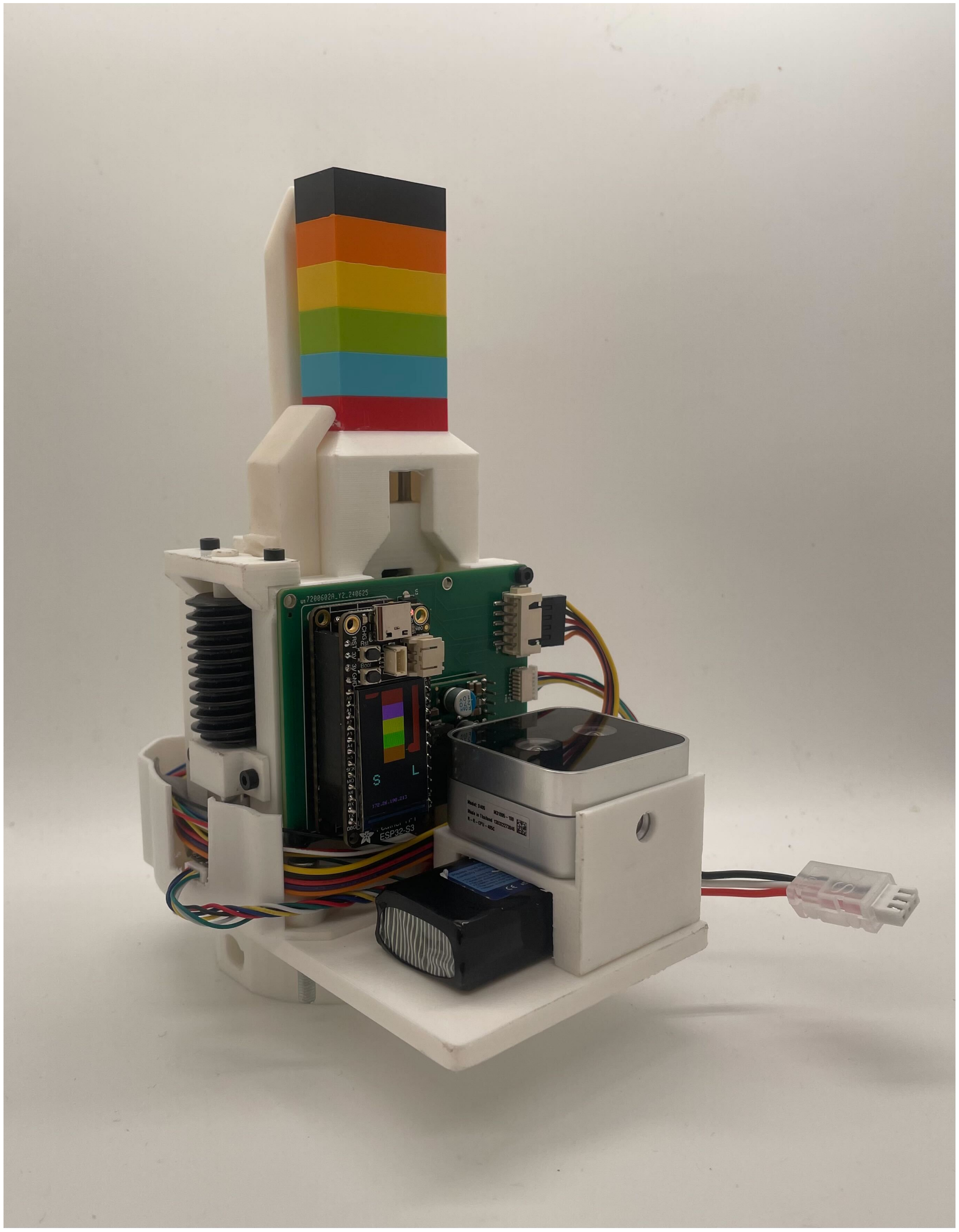


Figure 2.18: BrickPick-Tactile: Isometric View

2.3.2 Electrial

The electronics for BrickPick-Tactile are significantly improved in both footprint and reliability by creating a custom PCB. The PCB comes fabricated with a 3.3V buck converter, connectors for the motors, and free pins to connect the microcontroller, motor shield, and 6V buck converter. The TFT screen on the ESP32-S3 board is used to display information regarding the state of the end effector embedded software. When booted, the controller first connects to a local network with a network name and password saved during flashing. Following a successful connection to the network, the screen will display the IP address of the WiFi server along with PID targets marked with green hashes and digital fingers to show the estimated displacement of the moment plates.

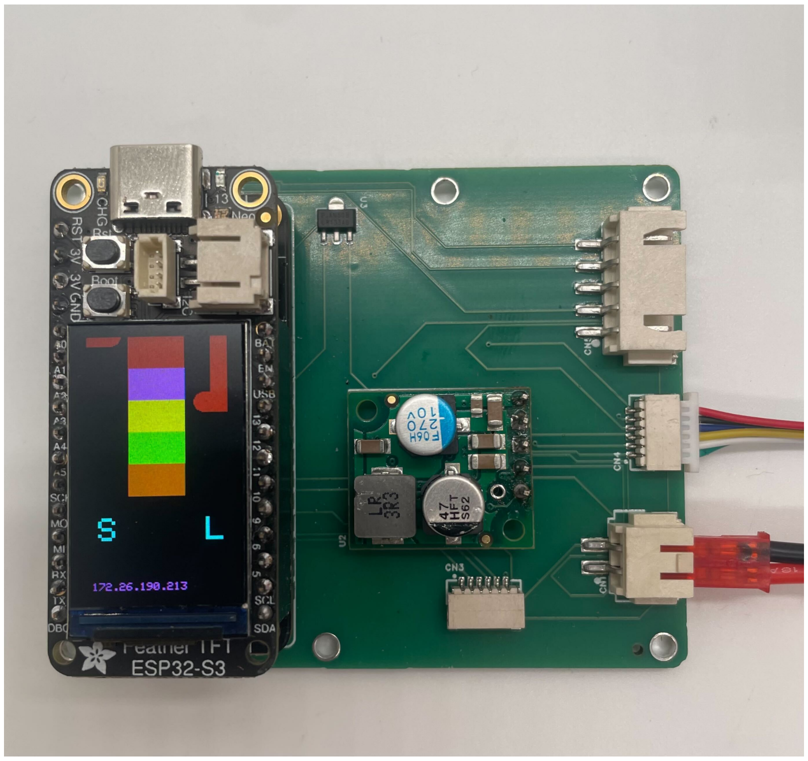


Figure 2.19: BrickPick-Tactile: PCB and Display

2.3.3 ROS Interface

BrickPick-Tactile is controlled in the same manner as BrickPick-Compact with HTTP GET requests. The python client adapter is now built into a ROS service node that will forward ROS service requests to the end effector through HTTP and respond to the ROS client with the HTTP reply. This allows for better integration into the ROS-based manipulation system used in the Robot Training Center. To control BrickPick-Tactile manually, a ROS teleop node is also created allowing the user to send commands through bound keyboard inputs as opposed to typed text commands.

Listing 2.2: BrickPick-Tactile: Teleop Interface

```
This node takes keypresses from the keyboard and publishes
   them as BrickpickCommand service requests
------------------------------
Velocity Control:
Stop: 'Space'
Reset: 'r'
Long plate:  /\\ Up
             ||
             \\/ Down

            Up   Down
Short plate: <===>

Position Control:
plunger     ('p', 'o')                     [Plunger down/up]
long plate  ('0', '1', '2', '3', '4', '5')   [Go to brick #]
short plate (')', '!', '@', '#', '$', '%')   [Go to brick #]

CTRL-C to quit
```

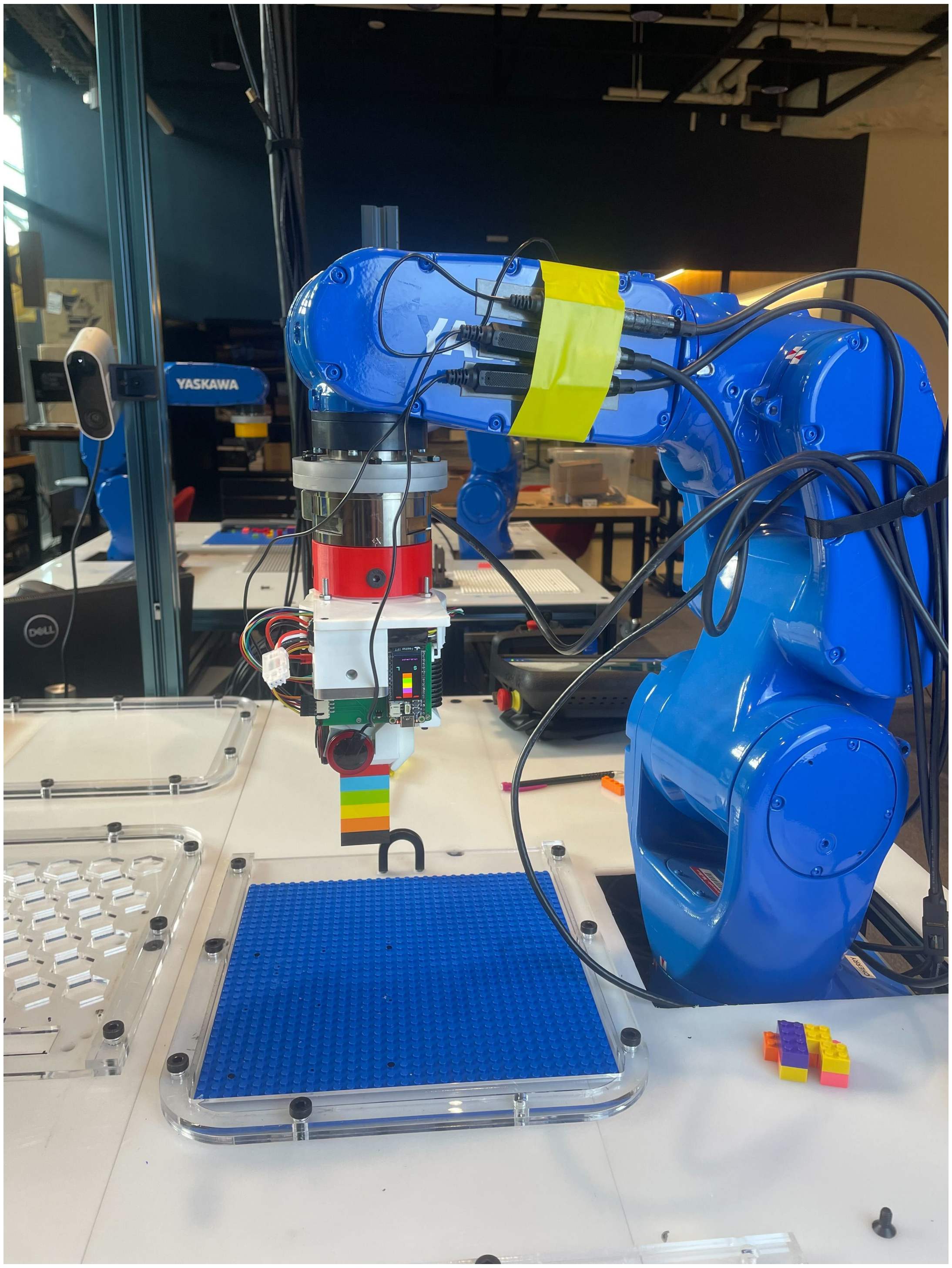


Figure 2.20: BrickPick-Tactile: Mounted on to YK-Creator

2.4 Comparison/Qualification

Figure 2.21 demonstrates the development in footprint and quality of the BrickPick end effector after each revision. The end effector is held in the same location for each photograph near its vertical limit to reach over the center of the build plate. BrickPick 2.21a nearly occupies the entirety of the vertical space available to the robot arm and requires the robot to exceed its joint space limits in order to perform picking rotations at certain locations on the build plate. BrickPick-Compact 2.21b has significantly more vertical clearance and permits the robot to perform pick rotations across the full span of the build plate. The large housing components require that obstacles in the work space are sufficiently low or distant from locations where the robot is operating. BrickPick-Tactile 2.21c has a similar vertical clearance to the prior iteration, but sports a significantly reduced horizontal footprint allowing for larger and closer workspace obstacles to exist near the end effector during assembly tasks.

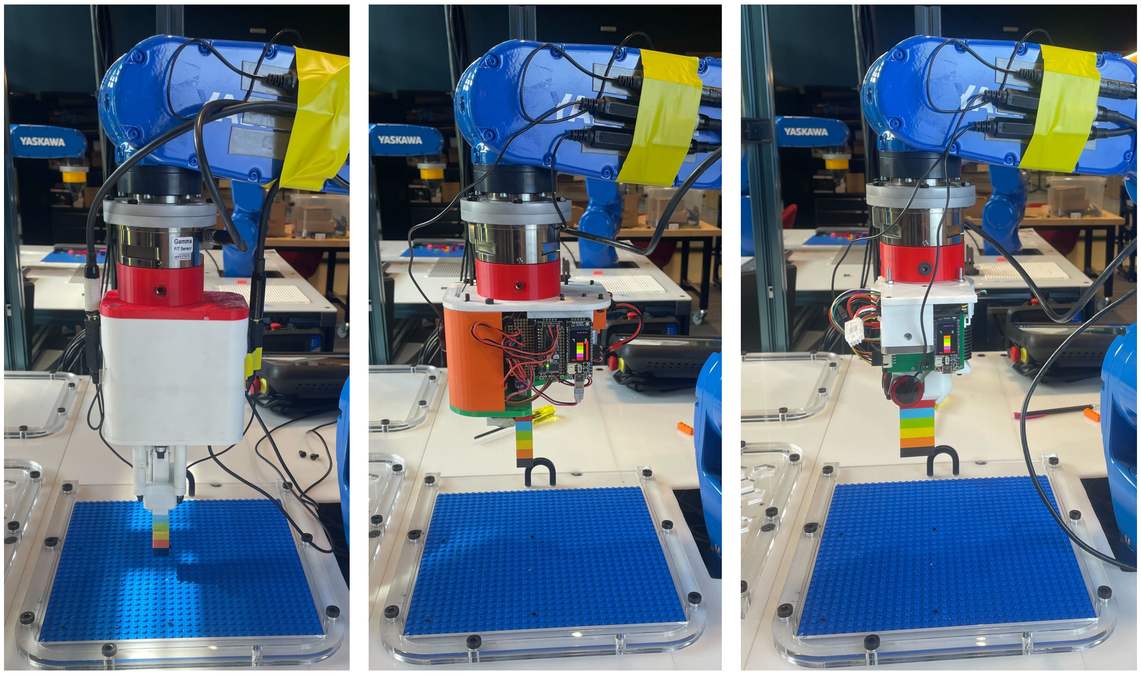

(a) BrickPick (b) BrickPick-Compact (c) BrickPick-Tactile

Figure 2.21: BrickPick: Footprint comparison mounted on to YK-Creator

Chapter 3

Vibro-Tactile Toolbox

Humans are often regarded as the most robust manipulation systems, owing to sensory feedback, intelligent planning, and a profound understanding of physical interactions. Our ability to seamlessly fuse information from all five senses to adjust and observe motor skills on the go allows for humans to achieve a level of dexterity far beyond the precision of our planned movements. For instance, placing a Lego block involves visual guidance to approximate its position and tactile feedback to press it into place, with lateral adjustments until it fits. This process inspires the fundamental research question: How can a robotic system utilize multi-modal feedback to enhance its dexterity and evaluate the outcomes of its manipulation tasks?

To address this question, we introduce the Vibro-Tactile Toolbox, a ROS-based manipulation system leveraging sensory feedback from the robot state, vision, force-torque sensors, and contact microphones for supervised skill execution. The toolbox comprises three primary modules: skill specification, outcome criteria, and termination criteria.

1. **Skill Module**: This module encapsulates 'skill-steps,' which pair robot actions with termination and outcome detection configurations. It also includes a policy for correcting or retrying failed actions. Currently random, this policy can be refined through collected data from trials where an action fails initially then succeeds in a retry.
2. **Terminator Module**: This module consists of termination handlers that stop

the robot when the specified termination criteria are met. These handlers monitor signals such as audio, force-torque, joint states, or time, determining whether the robot should cease its current action based on predefined thresholds or a trained policy. This functionality allows the robot to explore its action space autonomously without triggering any fail-safes that require human intervention.

3. **Outcome Detection Module**: This module includes outcome detection services that query sensory signals to verify if the skill has successfully achieved its task on termination. Initial data collection trials utilize visual or reaction force based outcome detectors in order to autonomously label training data for vibro-tactile outcome detection policies.

Before deploying neural network policies for skill correction, termination, and outcome detection, a labeled dataset is essential. While this is typically generated in simulation, most simulators cannot emulate vibrotactile data from collisions, vibrations, or geometric interference in an efficient and accurate manner. Collecting real-world data presents challenges, such as the loss of access to ground truth environment states and the absence of an 'env.reset()' command for catastrophic failures. Reinforcement learning, a prevalent method for training robot manipulation policies, necessitates exploration and failure to develop robust policies. Real robots will shut down and require human intervention to reset when catastrophic failures are experienced.

Our primary contribution is the development of a machine learning supervision framework in ROS, enabling a robot to autonomously explore failed actions with automatic termination and outcome labeling. Utilizing a threshold-based approach for termination and an off-the-shelf Detectron2 network to identify Lego bricks, we autonomously collect labeled data for Lego pick-and-place tasks with minimal human intervention.

3.1 Robot Training Center: Assembly Tasks

The Robot Training Center aims to study multi-level planning, execution and control for robotic assembly and disassembly. The Vibro-Tactile Toolbox is developed as a part of this project to provide a framework for skill execution, vibro-tactile feedback policies, and collecting data from robotic manufacturing tasks. Figure 3.1 shows a simplified view of the integration of major software modules operating within the RTC. The device layer contains all the necessary hardware drivers to abstract sensor signals into standard ROS message types which are published across the network. The skill layer are the code modules responsible for the control and execution of various robot actions outlined in a task script. The Vibro-Tactile Toolbox is primarily focused on this layer. The learn and execute module are the high-level task scripts responsible for planning robot actions to be attempted by the skill layer.

The data layer is responsible for the storage and distribution of data collected from assembly trials. This data can be saved directly to disk through ROSbag recordings or streamed to an MQTT listener socket via ROS messages. The toolbox also contains data post-processing scripts to extract ROSbag trial recordings into portable datasets. These datasets are split by robot skill, block type, volume, and arm velocity and will automatically extract a frame of data based on the recorded timestamp of a skill termination signal. The outcome label is applied to each snippet by finding the outcome detection message that corresponds to the recorded action.

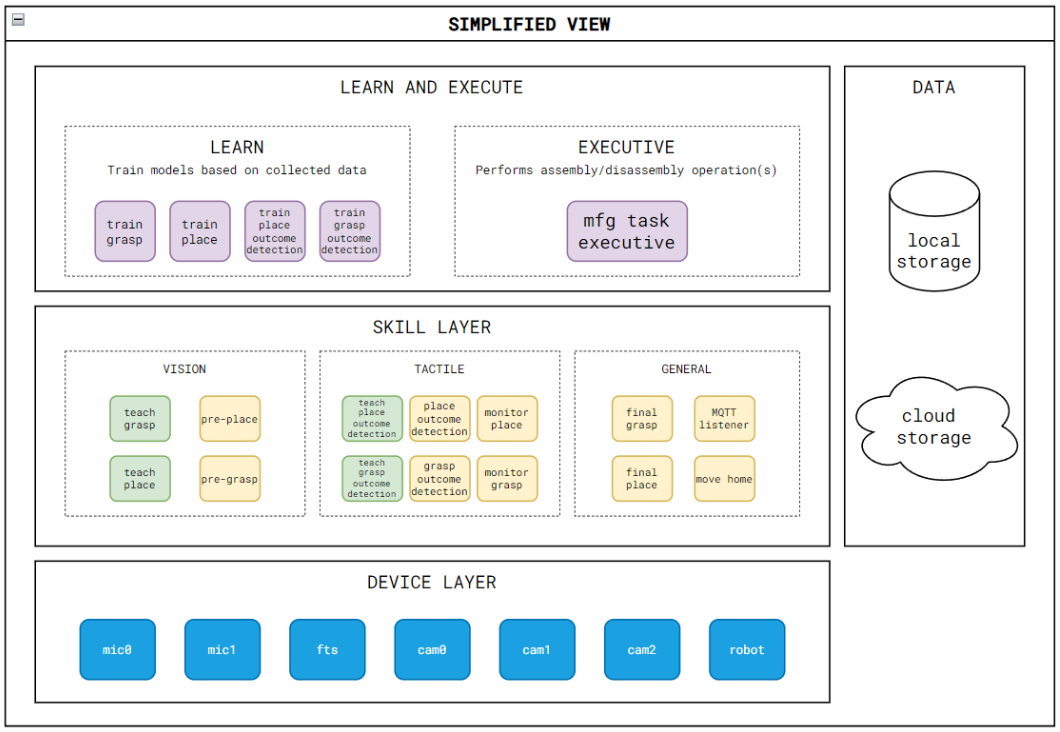


Figure 3.1: RTC Assembly: Simplified software modules.

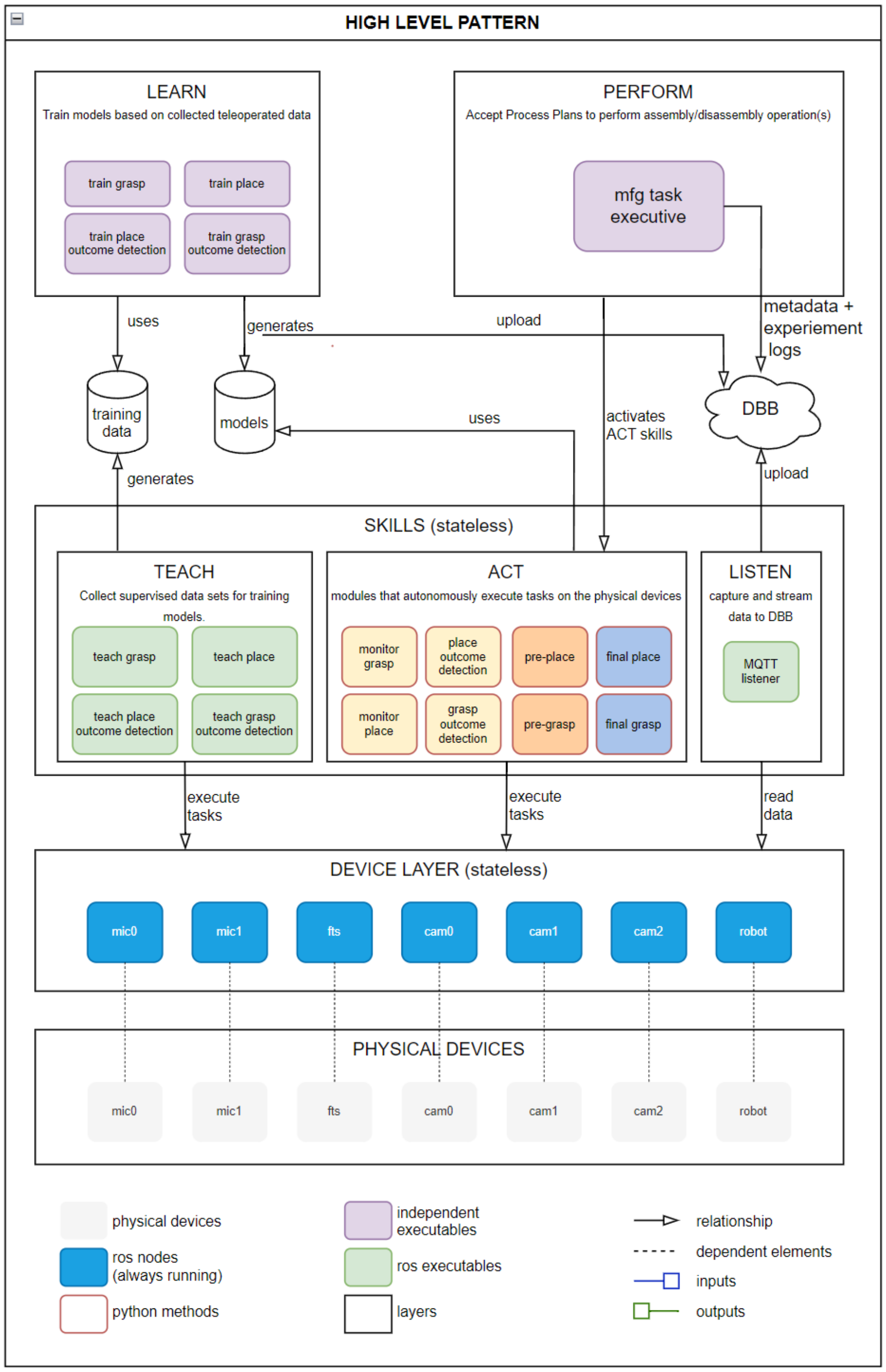


Figure 3.2: RTC Assembly: High level system diagram

3.2 Hardware Setup

Figure 3.3 shows our hardware setup at the MFI Robot Training Center using Yaskawa GP-4 manipulator named YK-creator. A studded build plate is slid into position by a human or robot with an Orbbec Femto Bolt side camera to monitor the robot's workspace. Attached to the end-effector is a piezeo-electric contact microphone, an Intel Realsense D405 stereo wrist camera, and a Robotiq Force-Torque Sensor (FTS). The Vibro-Tactile Toolbox interfaces with sensors through subscriptions to ROS topics which were readily available for off the shelf components. A custom audio device ROS publisher was developed to integrate the contact microphones into the system. Interfacing with the robot requires the creation of a 'robot_controller' module which allows the user to link robot hardware APIs to VTT robot commands. The end-effector used to develop the Vibro-Tactile Toolbox is the passive gripper discussed in Chapter 2.

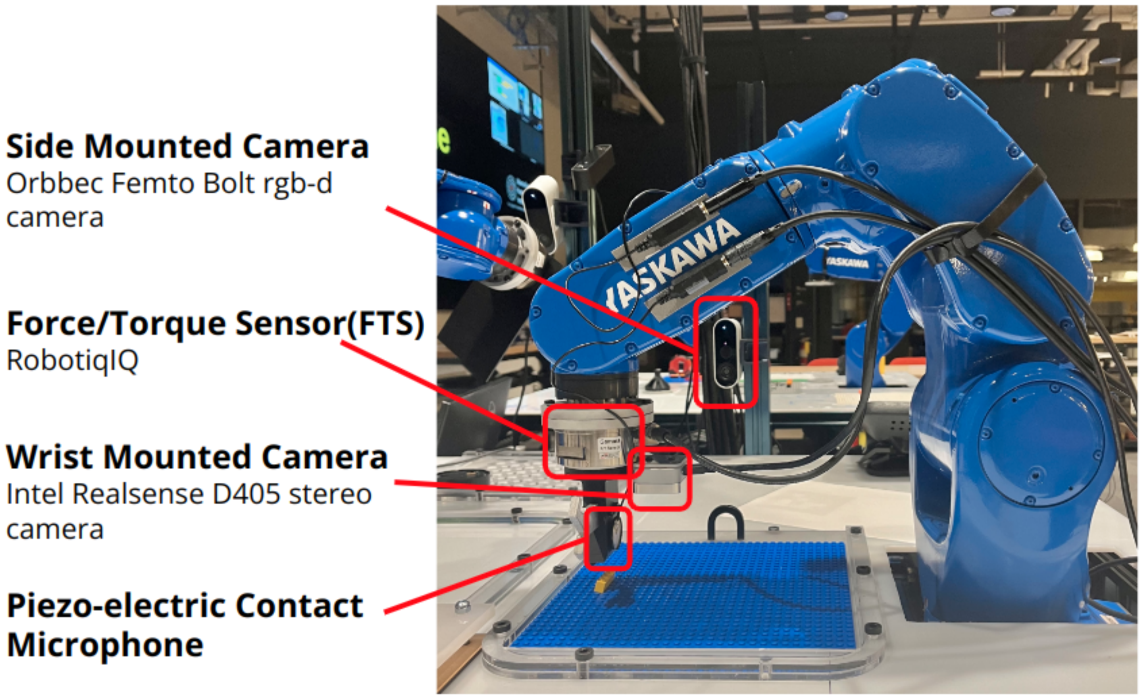

Side Mounted Camera
Orbbec Femto Bolt rgb-d camera
Force/Torque Sensor(FTS)
RobotiqIQ
Wrist Mounted Camera
Intel Realsense D405 stereo camera
Piezo-electric Contact Microphone
YASKAWA

Figure 3.3: Sensor setup on the Yaskawa GP-4 manipulator.

Two computers are used to manage the devices located in the RTC Mezzanine Lab: mfi-twin, the ’Primary Workstation,’ and yk-god, the ’Peripheral Devices Computer,’ shown in 3.4. All sensors are connected directly to the peripheral devices computer including cameras, force-torque sensors, microphones, and other miscellaneous devices. The robot arm controllers and both computers are connected through ethernet with a 1Gbs bandwith limit per channel. This proved to be a challenge for deployment, as raw 4k image frames published by the work space cameras are limited to a maximum of 5 frames per second to fully saturate the network switch. This means that video must be processed directly on the peripheral device computer or that the cameras must be used only for image capture when publishing to the primary work station. All robot controls and major software packages are deployed to the primary workstation, as it is a more powerful computer with a GPU to support real time ML inference. The only exception being the hardware drivers and vision-based outcome detector running on the PDC, publishing a short outcome detection service reply over the network instead of raw images.

The system is designed to integrate data collection with the Digital Backbone, a large data lake built to store and tag sensor data collected from robotic manufacturing tasks. The Digital Backbone is designed to be an archive of manipulation data that may allow researchers and engineers to access training data without needing to commit resources to collecting it independently. Currently data collected by trial runs with the Vibro-Tactile Toolbox is stored locally as ROSbag recordings and uploaded to the AVEVA PI archive device manually. As the project continues to develop, the goal will be to allow for real time publishing of ROS messages directly to the Digital Backbone through an MQTT Broker.

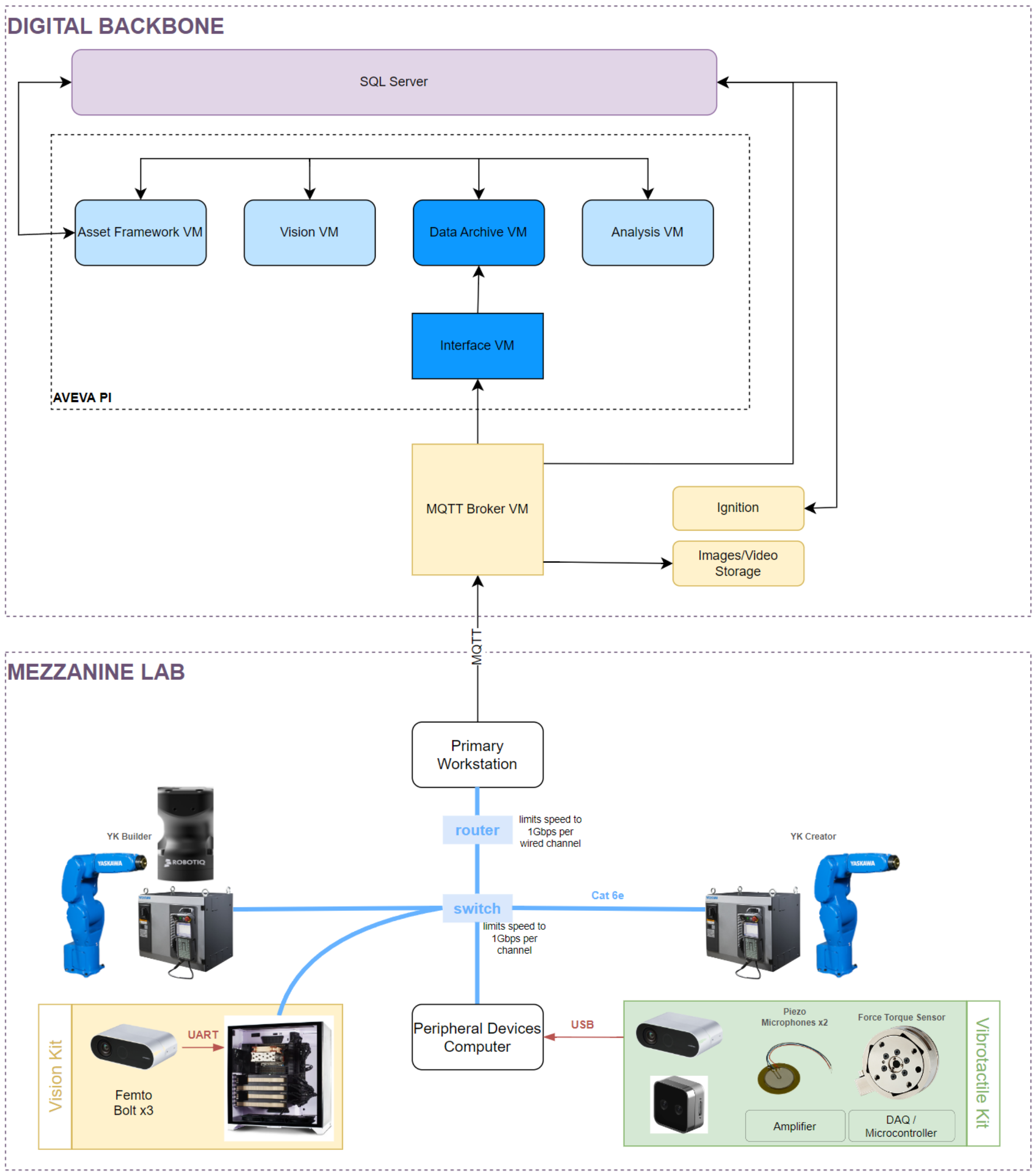


Figure 3.4: RTC Assembly: Hardware integration diagram

3.3 VTT System

The Vibro-Tactile Toolbox is designed to be a general framework that separates hardware requirements from its core using rostopics as an interface. Custom skills, terminators, and outcome detectors are implemented in a factory style to allow a user to easily develop custom modules to fit their use case. We use Lego brick assembly as an example task to demonstrate object pick and place skills in a setting where humans would utilize vibro-tactile feedback. Legos are a cheap and resistant analog for many important peg insertion tasks found in manufacturing and assembly such as joining electrical connectors, keyed shafts, and elastic bands.

The Vibro-Tactile Toolbox is comprised of three major modules: Skill Execution, Termination, and Outcome Detection. The system is designed to leverage hardware abstraction, requiring a developer to write a RobotCommander with a provided factory template as well as using standard ROS message types to receive sensor data. Generally, the terminator and outcome detector nodes subscribe to sensor rostopics and publish termination signals and outcomes back to the skill node. The skill node will execute a series of robot actions while listening for termination signals and making outcome service requests upon termination. When a skill is completed all of the sensor data and internal toolbox signals are recorded to a rosbag or streamed to an MQTT client through ROS messages over the network. Figure 3.5 shows the software stack used by the Vibro-Tactile Toolbox deployed at the RTC. The ROS Adapters/Drivers layer are hardware specific ROS nodes that publish or serve data through ROS topics. These are saved to a separate repository from the VTT called RTC-drivers. The ROS layer contains the modules that manage the execution and supervision of skills. This is where the robot controller, terminator node, outcome detectors, and skill execution node operate when a skill is launched. The Python script layer is where task scripts are run to plan the execution of a series of skills and other actions. A task script will call skills with different parameters to plan a series of pick and place actions for the robot to execute and optionally record. When recording, a task script will open and close a collector from the data collection layer in order to properly save or republish desired ROS topics.

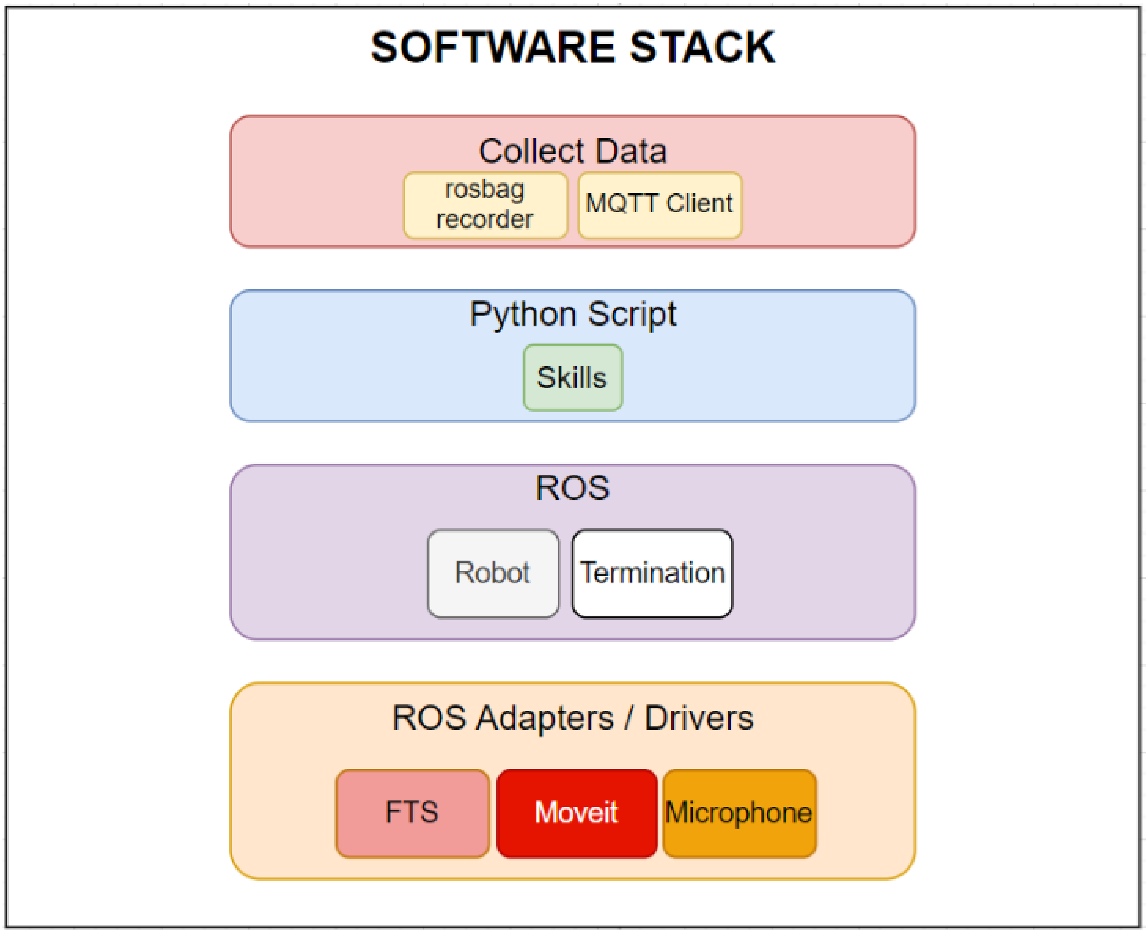


Figure 3.5: Vibro-Tactile Toolbox: Software stack with RTC hardware drivers. (Note Camera and Outcome Detector is missing from graphic)

3.3.1 Skill Execution

Skills are implemented by creating an ordered dictionary of skill steps, each containing lambda functions to produce the skill step name, robot command, termination configuration, and outcome request. When a skill is called for execution, each step is commanded in series. To execute a step, the toolbox will first publish the termination configuration to the terminator module to specify what sensor signals should cause the robot to stop. The robot is then commanded to act asynchronously until a termination signal is received. If an outcome detection function is specified, the toolbox will use the given function to request any outcome detection services to determine if the skill step has been completed successfully.

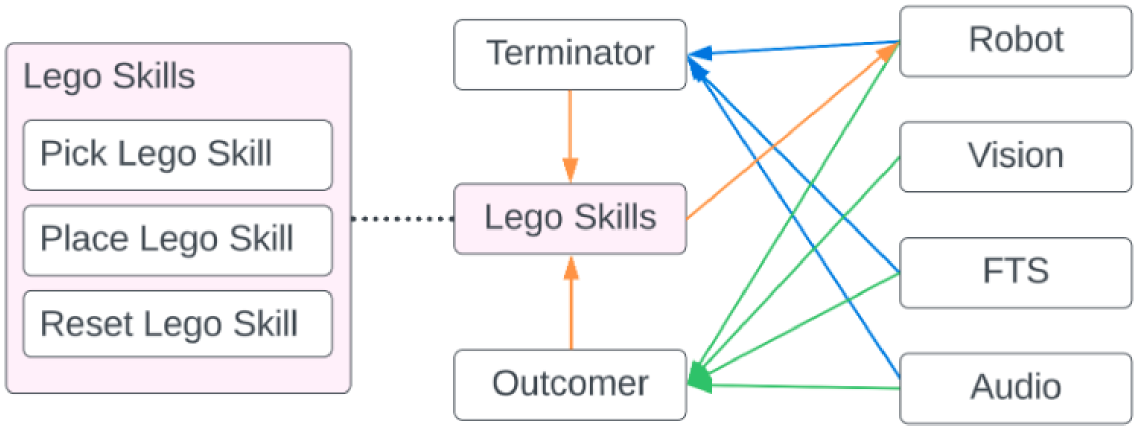


Figure 3.6: Vibro-Tactile Toolbox: ROS topic subscription diagram

Figure 3.7 outlines the full process diagram of a skill to be executed by the robotic assembly bed. The robot arm will first grasp the target object and determine the in-hand localization using TAX-pose, a task specific pose estimation system used for robotic manipulation developed by the Robots Perceiving And Doing lab [8]. TAX-pose then generates the estimated pose for insertion and passes the target pose to the Vibro-Tactile Toolbox. Using the estimated pick/place pose, the VTT begins executing the necessary sequence of skills to assemble the object. If a skill's outcome is a recoverable failure, the robot will try again with a new pose target to re-attempt insertion until successful. Once success has been observed the skill is complete and the system will move on to the next item in the task script.

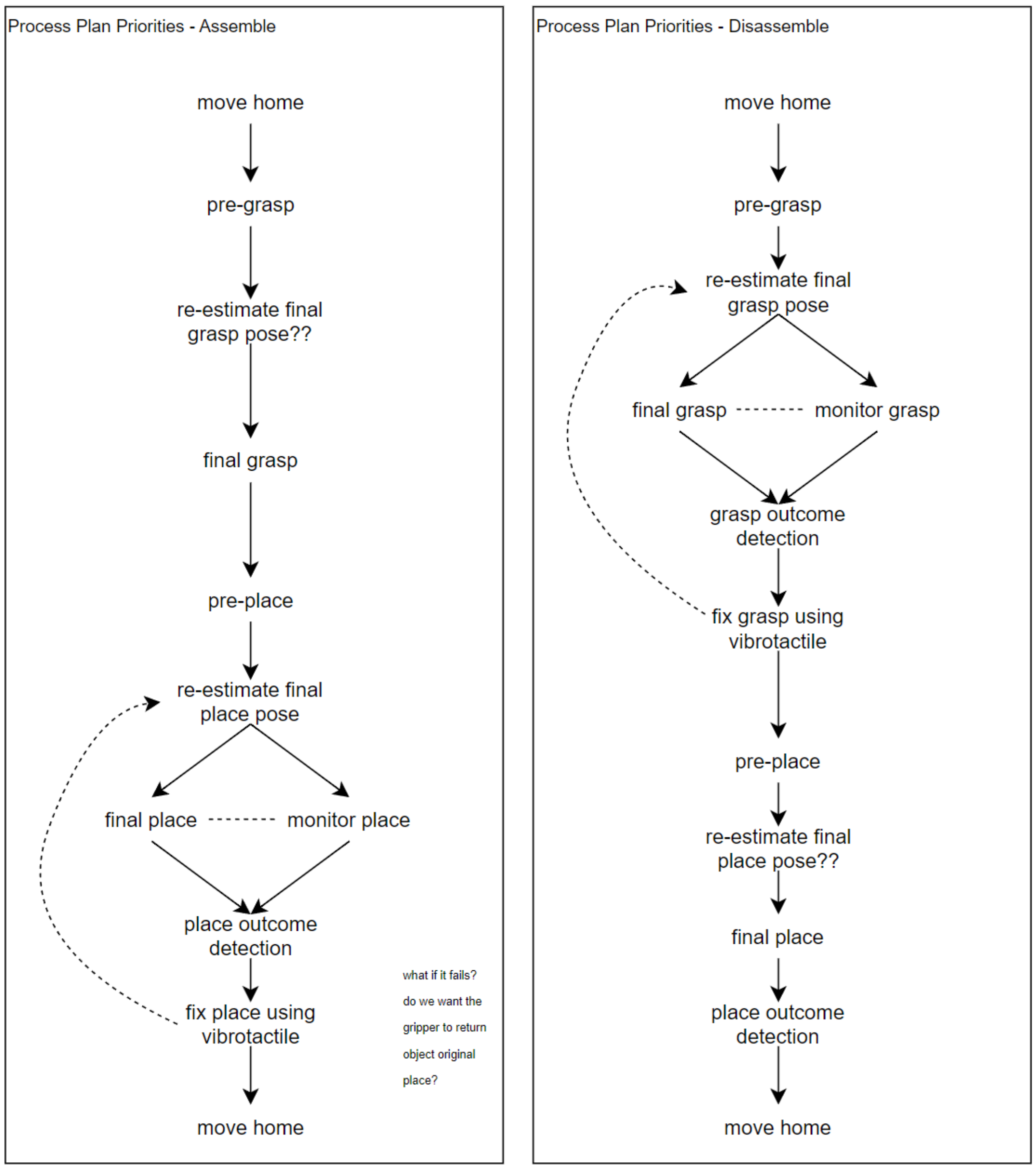

Process Plan Priorities - Assemble
move home
pre-grasp
re-estimate final grasp pose??
final grasp
pre-place
re-estimate final place pose
final place
monitor place
place outcome detection
fix place using vibrotactile
move home
what if it fails? do we want the gripper to return object original place?
Process Plan Priorities - Disassemble
move home
pre-grasp
re-estimate final grasp pose
final grasp
monitor grasp
grasp outcome detection
fix grasp using vibrotactile
pre-place
re-estimate final place pose??
final place
place outcome detection
move home

Figure 3.7: Vibro-Tactile Toolbox: Skill execution process diagram

Within the VTT, the robot controller, skills, and termination handlers are all implemented using a factory design pattern. An abstract base class for each is provided in the repository as a template for the required class methods. This is done to allow groups with different hardware to implement their own drivers and expand the capabilities of the system in a standardized fashion.

RobotController. An instance of a RobotController class is initialized with a type and name to save as metadata during data collection. It must have methods to determine the robot position, force-torque sensor reading, and commands to drive the robot to joint positions or end effector poses. It is important that the go to commands are non-blocking during execution, as the toolbox will handle termination and stopping. RobotControllers have been developed and tested on UR5e, Franka Panda, and Yaskawa-GP4 robot rams.

Skill. Skills are implemented as a list of individual skill steps which capture a single robot command with a corresponding termination configuration and an optional outcome report. For example, the PullUpSkill has two skill steps: 1. Move end effector up 10mm, terminate when reaction force is observed in downwards direction, outcome is successful if the change in the force is greater than the weight of the end effector. 2. Move end effector down 10mm, terminate when upward reaction force is observed, no outcome required. The skill determines if the end effector has been successfully connected to the target brick by measuring the reaction force between robot and build plate when pulled up, then re-engages the brick after its measurement. Skills are initialized with static parameters such as the end effector transform matrix and target pose for the skill. They can also be modified with other parameters such as perturbations or command velocity scaling in order to collect a wide variety of succeeded and failed skill trials.

Termination Handlers. Skill termination is done by having a set of termination handlers each listen to a single mode of data (audio, force torque, joint state, timeout, etc.) which all feed in to a master termination node. When a skill is executed it publishes a termination configuration to this master node which will update the parameters of each termination handler as well as setting which handlers it will listen to for a skill step. Better methods of skill termination can be added by implementing new TerminationHandler nodes from a standard ROS node and class structure.

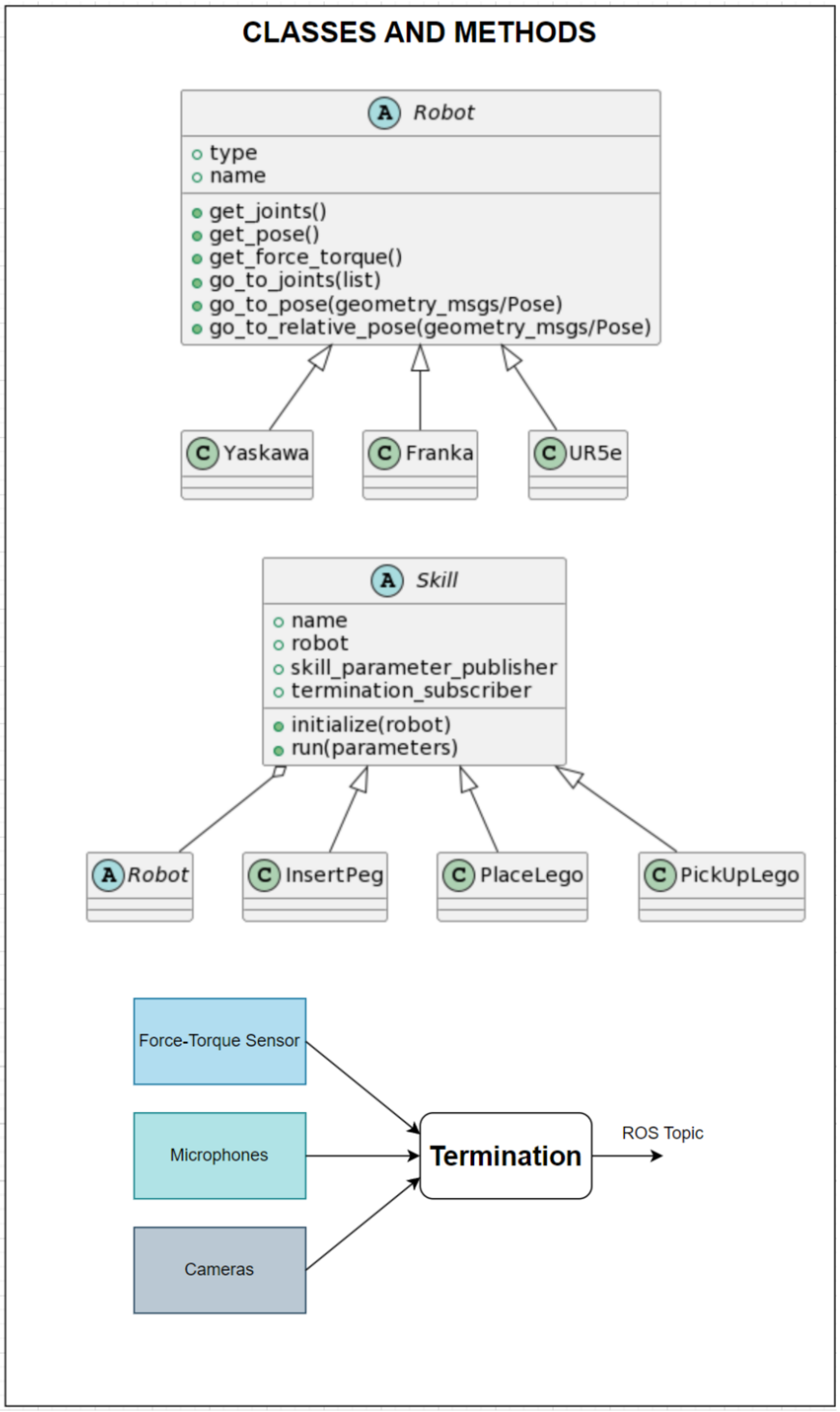


Figure 3.8: Vibro-Tactile Toolbox: Classes and methods

3.3.2 Skills

A skill in the Vibro-Tactile Toolbox is a series of skill steps each consisting of a step name, robot command, termination configuration, and outcome detection function. When a skill is called for execution it will step through every skill step, publishing the termination config, commanding the robot to move, terminating the action when conditions are met, and recording the outcome of the action. When the outcome of a skill step is a failure, a skill policy is used to re-attempt the step with new parameters. The skill policy is not yet implemented as more data is needed for training. To collect data, an open loop reset version of the Lego Place skill is written which takes a perturbation parameter to initially fail to make a connection, then resets and executes its action to the correct position.

MoveAboveLegoPose. This skill simply moves the end effector tip 10mm above the pose of the target Lego. It primarily uses the pose termination handler since it's just a rapid motion, but the termination config contains force-torque thresholds of $\pm 5N$ and $\pm 2Nm$ as safety thresholds in the event of a collision with workspace obstacles. This skill has an alternate version, MoveAbovePerturbLegoPose, which adds error to the commanded position in order to generate unsuccessful MoveDown trials.

MoveDown. This skill is meant to follow the MoveAbove skill. It drives the end effector down 12mm terminating when a reaction force between Lego and end effector is observed. The outcome of a successful MoveDown will result in a connection between end effector and target brick. This connection is verified and labelled using the PullUp skill.

PullUp. This skill is used to measure the clamp force between end effector and build plate. The robot is commanded to move up 10mm and terminates if the force-torque sensor measures a downwards reaction force. This skill uses the force-torque outcome detector to measure the force before and after moving upwards to label a prior MoveDown as successfully connected or not. After measuring the force, the robot moves back down to re-engage the build plate.

PickLego. To pick a Lego from the build plate the robot executes the following skill steps: 1. Rotate about target bottom edge towards the gripper's corner, terminate with force-torque or pose. 2. Remove brick by moving upwards, terminate on pose or timeout. The visual outcome detector is used to evaluate the success of this skill by determining whether the brick is stuck to the build plate or attached to the end-effector.

PlaceLego. The place skill for a lego using our end effector is nearly identical to the pick skill. Rather than rotating about the bottom edge of the target brick towards the end effector plate, for placing the robot will rotate about the top edge of the target away from the end effector plate. The pull up step is also used to record connectivity between the target and the build plate. The visual outcome now records a success if the target brick is disconnected from the end effector and attached to the build plate.

3.3.3 Termination

Termination. The terminator module is implemented using a master terminator node and a number of termination handler nodes. Each termination handler listens to sensor signals for a specific mode of termination outlined in table 3.1. The master terminator node is responsible for listening to all termination handlers and publishing a skill termination signal when an active termination handler signals for termination. When the skill execution loop publishes a termination configuration, this will specify which termination handlers are active during the execution of the next robot action as well as adding or changing parameters on the selected handlers.

Time, Joint, Pose, and Fts were used as the set of threshold based terminators to supervise initial data collection. Vision is not currently used for termination as no visual servoing skill exists yet. Following self-supervised data collection, NN policy based termination handlers can be trained to allow for contact classification and tactile servoing skills.

3.3.4 Outcome Detection

Outcome Detection The outcome detection module is implemented in a similar fashion to the termination handlers. A set of outcome detection services are available for the skill to query after terminating an action. Each outcome detector is uniquely implemented to determine whether a skill action has been successful based on a different set of sensor signals. Table 3.2 shows the implemented outcome detectors.

Termination Handler	**Parameters**	**Description**
Time	duration [s]	Terminate skill after a set timeout duration
Joint	position [rad] tolerance [rad]	Terminate skill after robot joints reach goal joints within a tolerance
Pose	position [m] pos_tolerance [m] orient_tolerance [rad]	Terminate skill after robot ee frame reaches goal pose within a tolerance
Fts	check rate [ns] threshold [N, N-m]	Terminate skill after force torque sensor exceeds threshold
Audio	Not implemented	Terminate skill after audio signal is classified as a 'terminal contact'
Vision	Not implemented	Terminate skill after cameras detect a 'terminal visual state'

Table 3.1: Termination Handlers

The force-torque outcome detector is used to determine if a Lego brick is connected to the build plate by measuring the difference in fts readings before and after pulling up with the end effector. The vision outcome detector is a fine-tuned detectron model that can segment Lego bricks which is used to visually inspect bricks for connectivity. These two detectors are used to supervise/label skill outcomes for the initial dataset which can be later used to train NN policy based outcome detectors for audio or multi-modal inputs. An audio outcome detector was trained using a CNN acting on the audio spectrogram, but more data is required to improve its performance to a usable level.

3.4 Data Collection

3.4.1 Data collection

Data collection is done by creating a task script to launch the vibro-tactile toolbox core as well as using a rosbag recorder to capture relevant data during skill executions. When a skill is executed the sensor inputs, robot state, skill parameters, terminals,

Outcome Detector	Parameters	Description
Fts	threshold [N, N-m] first [bool]	Detect a successful connection by measuring the change in fts reading first after the skill step then second after pulling away.
Vision	n_bricks [int]	Detect a successful connection by counting if the detected brick segments matches the expected count. 1 for connected, 2 for disconnected.
Audio	None (NN policy)	Classify an audio signal as a successful connection or other contact event.

Table 3.2: Outcome Detectors

and outcomes are all recorded into a rosbag. The recorded rosbags then parsed into datasets containing compressed time synchronized data for later replay. Each skill trial can be displayed with a Qt window with a time step slider for playback.

Rosbag parsing. Recorded rosbags save all raw messages into a single file which can be very space inefficient for audio and video data. To produce a portable dataset from skill trials, rosbags are parsed into dataset folders containing information from the vibro-tactile core and sensors. Camera data is saved as .mp4 files to significantly reduce the memory footprint of video. Audio from contact microphones is saved as a .wav file for similar reasons. The robot state and force torque data are stored as numpy time series arrays. Termination signals and outcomes are saved as text files which record the timestamp, cause, and path to other relevant files (such as an annotated bounding box image for a visual outcome detector). The parsed datasets can be played back using a Qt window that will playback the rosbag with all relevant data plotted/displayed. Data in this format can be easily loaded into a machine learning framework such as Torch to use collected data to train vibro-tactile policies.

Registration. The position of the Lego build plate is recorded by using kinematic registration to locate studs on the build plate in the workspace of the robot. Prior to running skill execution, a human operator must manually drive the robot into the build plate to register a pose for the build plate origin.

Initial data collection. In order to train policies used in vibro-tactile skill

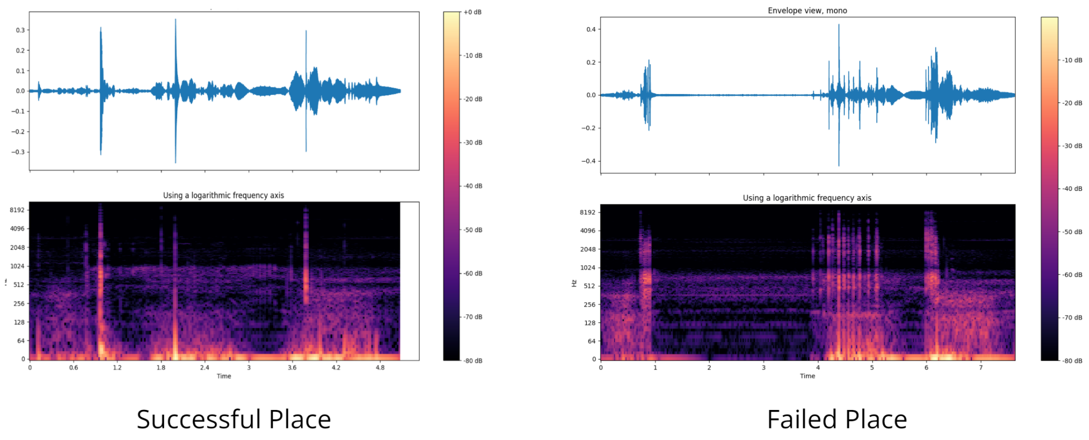


Figure 3.9: Audio signal visualization with Lego making contact on the table.

execution, an initial dataset of failed and successful skills must be collected. To produce this dataset, we use a task script that will repeatedly run the ”Lego Place (Open Loop Reset)” skill with a new perturbation each trial. This skill applies a perturbation to the ”Lego Place” skill on its first attempt, then retries the skill with no added perturbation. By using an open loop reset, we can ensure that our training dataset has explored a wider distribution of both failed and succeeded skill actions. A threshold based force torque terminator/outcome detector and bounding box based visual outcome detector are used to supervise each skill trial.

3.5 Results

3.5.1 Initial results

Our preliminary experimentation with the Vibro-Tactile Toolbox demonstrates its effectiveness in supervising autonomous data collection on real hardware without the need for human intervention. We observe audio peaks as the Lego makes contact with the table when the robot presses down.

3.5.2 Autonomous Data Collection

The Vibro-Tactile Toolbox successfully supervised autonomous data collection on real hardware, demonstrating its capability to handle real-world robotic tasks without human intervention. Using threshold-based termination criteria, the system effectively managed to prevent catastrophic failures, proving its reliability and robustness in a dynamic environment. The initial data collected includes various instances of both successful and failed skill executions, providing a rich dataset for further analysis and policy training.

3.5.3 Effectiveness of Termination Handlers

The implementation of the termination handlers proved effective. For instance, the force-torque sensor termination handler responded accurately by stopping the robot when the force exceeded predefined thresholds, which prevented any potential damage to both the robot and the environment. Similarly, the joint and pose termination handlers demonstrated precision by halting the robot's actions within the desired tolerances.

3.5.4 Outcome Detection Accuracy

Initial tests of the outcome detectors indicated reasonable accuracy. The force-torque outcome detector successfully identified connection strengths based on changes in force measurements, while the visual outcome detector effectively used detectron-based segmentation to classify connections between Lego bricks. However, there were instances of misclassification, particularly when the robot did not pull up far enough, suggesting room for improvement in sensor sensitivity or algorithm adjustment.

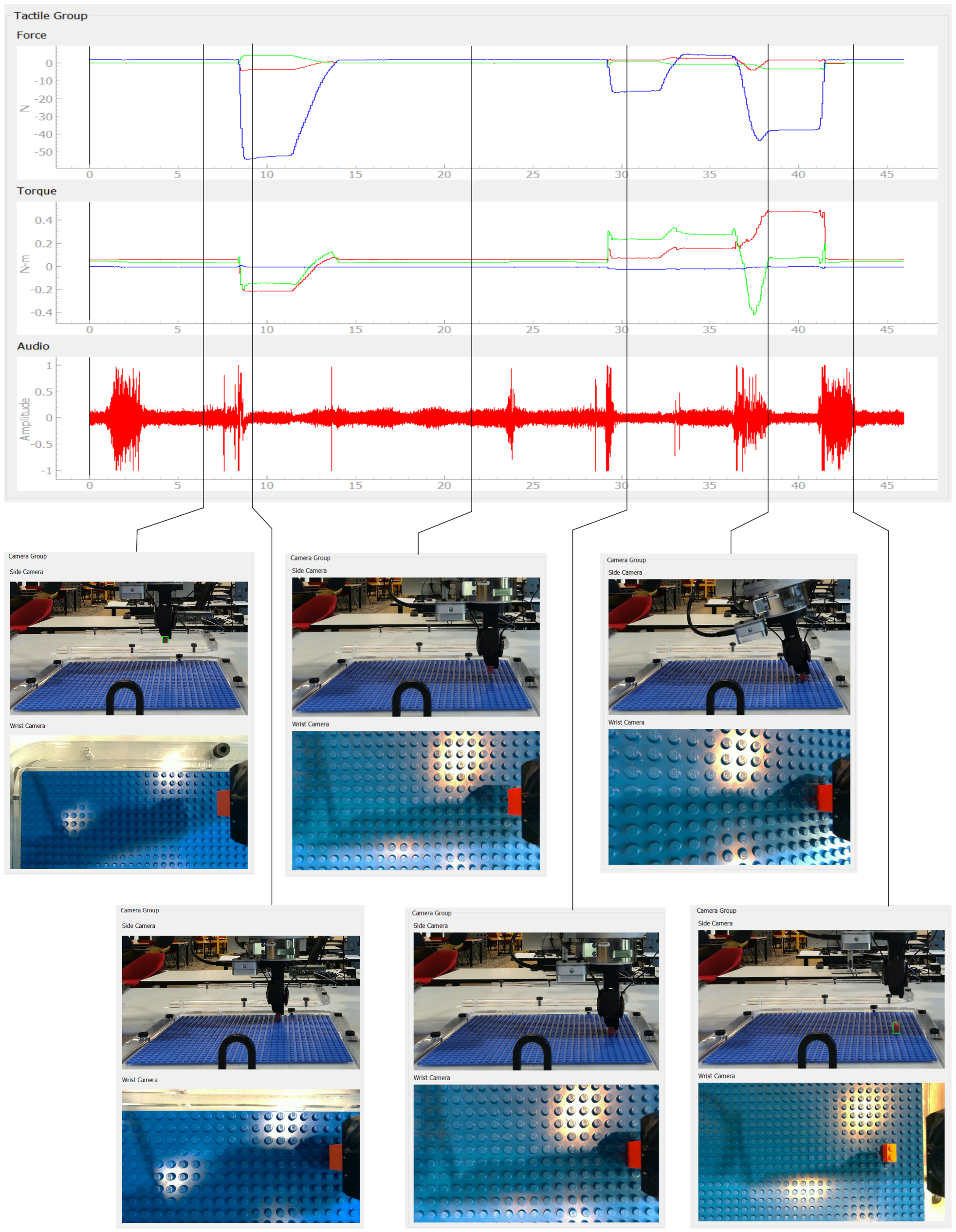


Figure 3.10: A successful LegoPlaceSkill data collection rollout. Critical events in order are: Approach above target, Engage brick (with error perturbation), Pull Up to check connectivity (Outcome: FAIL), Engage brick (no perturbation), Pull Up to check connectivity (Outcome: SUCCESS), Place Lego with rotation, Disengage brick (Outcome: SUCCESS).

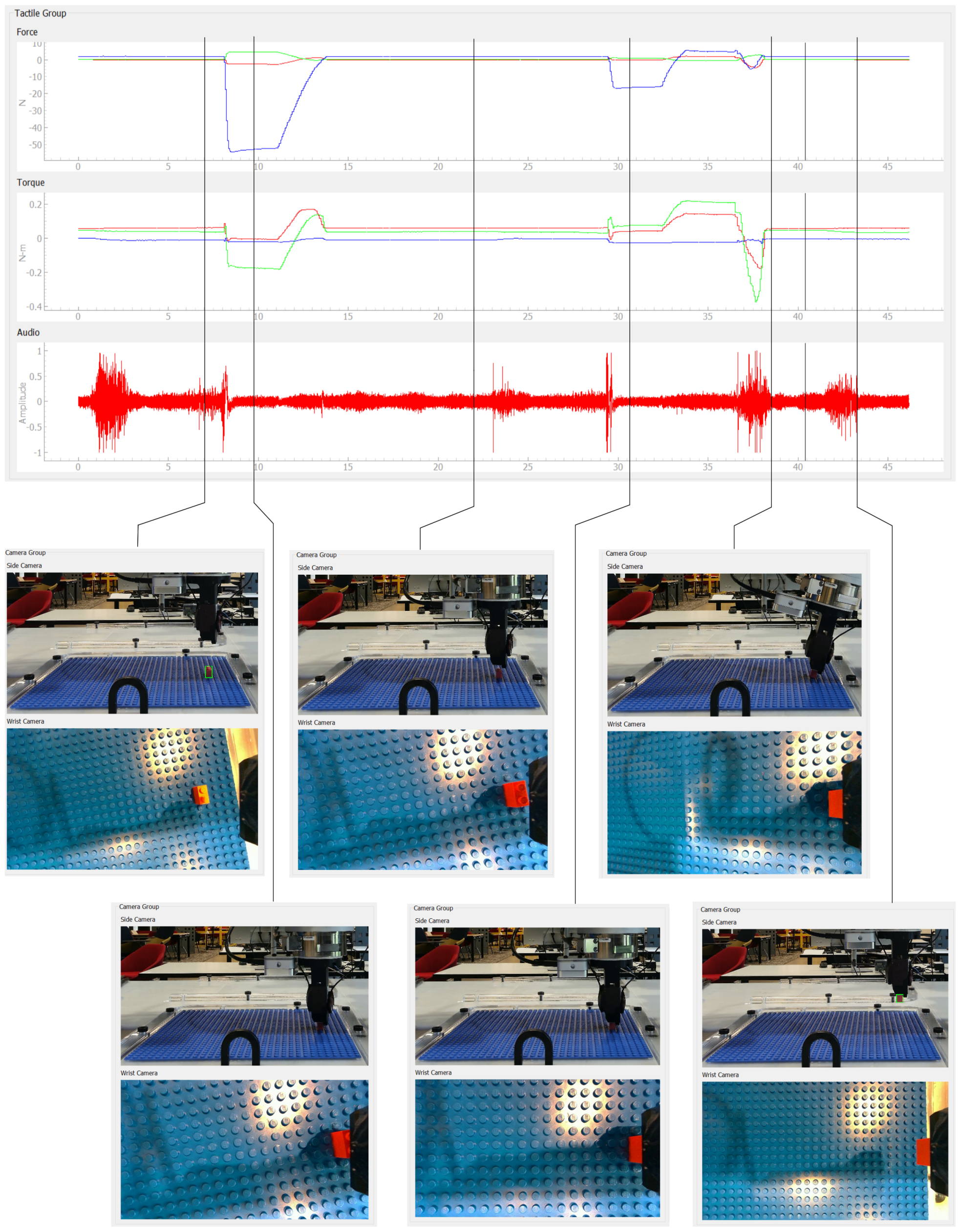


Figure 3.11: A successful LegoPickSkill data collection rollout. Critical events in order are: Approach above target, Engage brick (with error perturbation), Pull Up to check connectivity (Outcome: FAIL), Engage brick (no perturbation), Pull Up to check connectivity (Outcome: SUCCESS), Pick Lego with rotation, Disengage brick (Outcome: SUCCESS).

Chapter 4

Conclusions

4.1 BrickPick End-Effector

The development of the BrickPick end effector marks an advancement in efficiency and operability of Lego block manipulation, showcasing the iterative design and engineering process. Prior works leveraged an existing Lego brick separator tool, which provided valuable insights into the fundamental strategies for picking and placing Lego blocks. The primary features, such as the stud connection and corner rotation served as a foundation for future iterations by establishing the connect and rotate strategy.

The introduction of the BrickPick prototype validated the concept of using extendable moment plates to increase payload capacity. However, the challenges associated with torque-induced buckling during placement were identified, leading to the incorporation of a plunger mechanism. This enabled the precise placement of Lego stacks without compromising structural integrity.

The second revision, BrickPick-Compact, addressed practical issues such as size reduction and integration with smaller robotic arms. By replacing the linear actuators with a screw-rack mechanism and incorporating a wireless interface, the design became more compact and suitable for deployment in real-world assembly tasks. The addition of a wrist-mounted camera and battery further streamlined the setup, reducing external cabling and enhancing operability.

The final iteration, BrickPick-Tactile, focused on refining the electronics and overall design. The replacement of soldered prototype electronics with a printed circuit board (PCB) and the reduction of part count significantly improved reliability and manufacturability. This polished prototype demonstrated enhanced functionality, reducing workspace constraints and ensuring consistent performance in assembly tasks. A number of key ideas emerged during the development of BrickPick.

1. **Adaptation of Existing Tools:** Utilizing existing tools, such as the Lego brick separator, provided a valuable starting point and informed the initial design decisions.
2. **Iterative Design:** Continuous refinement based on testing and feedback was crucial for addressing challenges and improving performance.
3. **Size and Integration Considerations:** Designing for compactness and integration with smaller robotic arms and sensors ensured the end effector's suitability for practical applications. The size limitations of the end effector were far more profound when attempt to use it on hardware compared to modelling in CAD.
4. **Electronics and Control Enhancements:** Advancements in electronics, including the transition to a PCB and the implementation of wireless communication, played a critical role in achieving a reliable and user-friendly end effector.

In conclusion, the BrickPick end effector represents a solution for efficient and reliable Lego manipulation. Its development highlights the importance of iterative design and thoughtful integration, providing valuable insights for future advancements in robotic manipulation and manufacturing processes.

4.2 Vibro-Tactile Toolbox

Through the course of this project we have developed a real world manipulation supervisor that can inform the termination and outcome of a robot action. Our preliminary results show that the Vibro-Tactile Toolbox can successfully supervise a round of autonomous data collection on real hardware without the need for human intervention. Training vibro-tactile policies on the collected data has not yet been successful, but we're optimistic that these policies can be made to work with more data and experimentation with network architecture.

Threshold driven termination. To collect our initial dataset, threshold driven termination handlers were used to provide basic supervision capabilities over robot actions. Pose and joint termination handler were used with a tolerance matching the reported precision of the Yaskawa GP-4 robot (1mm, 0.01 rad). The force torque terminator used a typical threshold of 40N to prevent the robot hard stops from activating (120N). For robot actions that would engage the end effector with the build plate, a more precise threshold of 30N in the vertical axis was used to ensure a reasonable fit between Lego bricks. These termination handlers were sufficient to protect the robot from requiring a human reset when the end effector would fail a connection and attempt to drive though the build plate. The threshold based termination handlers provide the Vibro-Tactile Toolbox with standard modes of terminating robot actions as well as safety stops to prevent catastrophic failure. These are the primary requirements for collecting data on real hardware autonomously.

Initial outcome detection. The outcome detectors used for initial data collection were a threshold based force-torque detector and a bounding box based visual outcome detector. The force-torque outcome detector was designed to be requested twice. First after the robot engaged the end effector with the build plate, and second after the robot pulled up from the target brick. This outcome detector was used to determine if the measured change in force along the vertical axis indicated a tight connection between end effector and build plate. The visual outcome detector uses a fine-tuned detectron model to segment Lego bricks in the robot workspace. The detector will predict bounding boxes for each segment of lego brick(s) in the image frame near the end effector. If two boxes are detected, the bricks are labeled as disconnected, and if only one is detected the bricks are labeled as connected. There is much more room for improvement for this detector, as it will need to segment all bricks individually to function well on more complex build tasks. These two outcome detectors were reasonably accurate for labeling trial success, with a few common modes of failure. At times, the robot would not pull up far enough on a brick resulting in the fts outcome to be disconnected, as the force never crossed zero to indicate resistance. The visual outcome detector would also misidentify Lego bricks causing a trial to be labeled incorrectly. Our preliminary outcome detection module allowed us to collect our initial dataset with reasonably accurate self-supervised labeling.

Initial data collection. Our initial set of trial data was collected using threshold-based termination and labeled with our initial outcome detectors. This trial run demonstrates the self-supervising capabilities of the Vibro-Tactile Toolbox. The skill used to collect data was a Lego Place skill with an open loop retry. For each trial, a perturbation was applied to the open loop retry skill. This would execute the normal Lego Place skill registered directly to the build plate with the perturbation applied on its first attempt to engage the build plate. On its second attempt the perturbation was removed, resulting in an action that would fail on its first attempt and succeed on its retry. This produced an autonomously collected and labeled dataset containing an equal split of vibro-tactile data for failed and successful skill actions. The data collected by this trial run was also used to fine tune the actions and termination configurations for the lego skills.

Training results. Following the collection of our initial dataset, we tried training a neural net policy for audio outcome detection. The network was a multilayered Convolutional Neural Network (CNN) that would operate on a spectrogram of the past 1 second of audio following a skill termination signal. We were unable to get the audio detector to achieve a usable performance at the time of this paper, as it would output a successful label on nearly every attempt. More trial data and experimentation with network structure is required to take advantage of the available vibro-tactile data.

Future work. Now that we have established a framework for supervised data collection, we can begin experimenting with policy based skill execution, termination, and outcome detection. Using the initial dataset, we will attempt to train networks to use both audio and force-torque data to classify contact events. For Lego building, these classifications would be no-contact, misaligned contact, incomplete contact, and complete contact. More work can be done to improve the performance of the segmentation based visual outcome detector, as it currently operates on more assumptions than necessary about workspace outcome. The end goal of the project is to use the toolbox to train a skill execution policy that can fail and successfully retry an action using closed loop vibro-tactile feedback. Performance could also be improved by attempting to implement visual and tactile servoing to allow the robot to 'search' for a correct Lego peg insertion prior to engaging a brick.

Bibliography

[1] David V. Gealy, Stephen McKinley, Brent Yi, Philipp Wu, Phillip R. Downey, Greg Balke, Allan Zhao, Menglong Guo, Rachel Thomasson, Anthony Sinclair, Peter Cuellar, Zoe McCarthy, and Pieter Abbeel. Quasi-direct drive for low-cost compliant robotic manipulation, 2019. URL https://arxiv.org/abs/1904.03815. 1.4

[2] The LEGO Group. Lego® classic brick separator, 2024. URL https://www.lego.com/en-us/service/brickseparator/classic/. (document), 2.1

[3] S. Hutchinson, G.D. Hager, and P.I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, 1996. doi: 10.1109/70.538972. 1.4

[4] Rafał Kluz and Tomasz Trzepieciński. The repeatability positioning analysis of the industrial robot arm. *Assembly automation*, 34(3), 2014-7-29. ISSN 0144-5154. 1.4

[5] Moonyoung Lee, Kevin Zhang, George Kantor, and Oliver Kroemer. Contact classification for agriculture manipulation in cluttered canopies. February 2022. 1.3.4

[6] Jingshu Liu and Yuan Li. An image based visual servo approach with deep learning for robotic manipulation. *CoRR*, abs/1909.07727, 2019. URL http://arxiv.org/abs/1909.07727. 1.4

[7] Ruixuan Liu, Yifan Sun, and Changliu Liu. Robotic lego assembly and disassembly from human demonstration, 2023. URL https://arxiv.org/abs/2305.15667. 1.3.2

[8] Chuer Pan, Brian Okorn, Harry Zhang, Ben Eisner, and David Held. Tax-pose: Task-specific cross-pose estimation for robot manipulation, 2024. URL https://arxiv.org/abs/2211.09325. 3.3.1

[9] Peter Pastor, Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, and Stefan Schaal. Skill learning and task outcome prediction for manipulation. In *2011 IEEE International Conference on Robotics and Automation*, pages 3828–3834, 2011. doi: 10.1109/ICRA.2011.5980200. 1.4

[10] Stephen Smith, Oliver Kroemer, Jiaoyang Li, Zachary Rubinstein, Zeynep Temel, and Norman Sadeh. Multi-level planning, execution and control for robotic assembly and disassembly, 2024. URL `https://engineering.cmu.edu/mfi/research/projects/multi-level-planning-execution-control.html`. (document), 1.3.1, 1.1

[11] Yu Sun, Joe Falco, Máximo A. Roa, and Berk Çalli. Research challenges and progress in robotic grasping and manipulation competitions. *CoRR*, abs/2108.01483, 2021. URL `https://arxiv.org/abs/2108.01483`. 1.4

[12] Kevin Zhang, Mohit Sharma, Manuela Veloso, and Oliver Kroemer. Leveraging multimodal haptic sensory data for robust cutting. *CoRR*, abs/1909.12460, 2019. URL `http://arxiv.org/abs/1909.12460`. 1.3.3