Multiply Labs Robotics Software Technical Challenge

How does this challenge work?

The challenge is composed of two parts:

1. The “take-home” technical assignment (this challenge)
    - You should aim to complete this part within 4 hours and email us back your response as soon as you are done.
    - All of the information you should need for this challenge is included in this interview packet.
    - If you need to make any assumptions about this challenge, please state what assumptions you made.
    - We don’t expect you to fully complete every part of the challenge. Prioritize working on whichever parts you feel will best show off your strengths!
2. Multiply Labs internal review of your technical challenge
    - We will review your technical challenge. This review will consist of the following:
        - Reviewing your implementation of the questions.
        - Reviewing how you define your data structures.
        - Reviewing how you test your code to ensure it works as expected.
        - How intuitive it is to run your code and call specific functions.
        - We will actually run your code, which may include us calling specific functions you define.
        - **Make sure your code is runnable!**

You must use python for this challenge (unless otherwise stated).

You will submit:

- A zip file that contains all of the code you wrote for this challenge.

Good luck with the challenge and have fun!

Challenge:

You may use all of the contents in this zip folder as well as any information in the rest of this document to help you answer the questions below. **See the write-up below for more information on how the system works.**

Questions:

You are required to answer Questions 1, 2, and 3. If you have additional time, feel free to work on Question 4 or Question 5.

Question 1:

Write a client that can communicate with the dashboard of the robot, turn on the robot, and release the brakes. These steps must be completed before we are able to move the robot.

Supplemental Material:

- Please see the website from Universal Robotics that explains the dashboard server here: https://www.universal-robots.com/articles/ur/dashboard-server-e-series-port-29999/
    - Scroll down to the bottom of the page and download the PDF that explains the commands for this server
    - If downloading this PDF from the website does not work, the file "dashboard_server.pdf" is included in this zip file.

Question 2:

Design a driver that can pick and place the GRex from one location to another location in the cluster without hitting any known obstacles.

Supplemental Material:

- File: "robot_communication.py"
    - This file contains helper functions that you may use to control the robot.
- File "grex_location_data.csv":
    - This file contains information regarding the locations the GRex can be picked and placed from in the robotic cluster
- File "module_data.csv":
    - This file contains information regarding the width of the modules within the cluster

Additionally, see the writeup below for more information on how the GRex is moved from one location to another.

Question 3:

Incorporate the Dashboard Client with the code written for picking and placing a GRex from one location to another. The code should be able to know if the robot is in a state to interact with objects. If it is not in this state, it should turn the robot on as described in Question 1.

Question 4:

Add in the ability to recover from faults when the arm hits an unknown piece of hardware.

Question 5:

Design and implement a system that monitors and logs the robot's performance metrics, such as joint positions, velocities, and applied forces, during the pick-and-place operation. The system should also detect any deviations from expected values and initiate corrective actions or safety stops if required. Ensure that the system accounts for potential hardware issues, such as joint overloading or unexpected resistance, and integrates with the fault recovery mechanism from Question 4.

Background Information:

About the System:

Our system uses a 6DOF robotic arm on a rail to move bio materials between subsystems, which we call modules.

You can find more information of our system here:

- https://multiplylabs.com/products/cell-therapy/

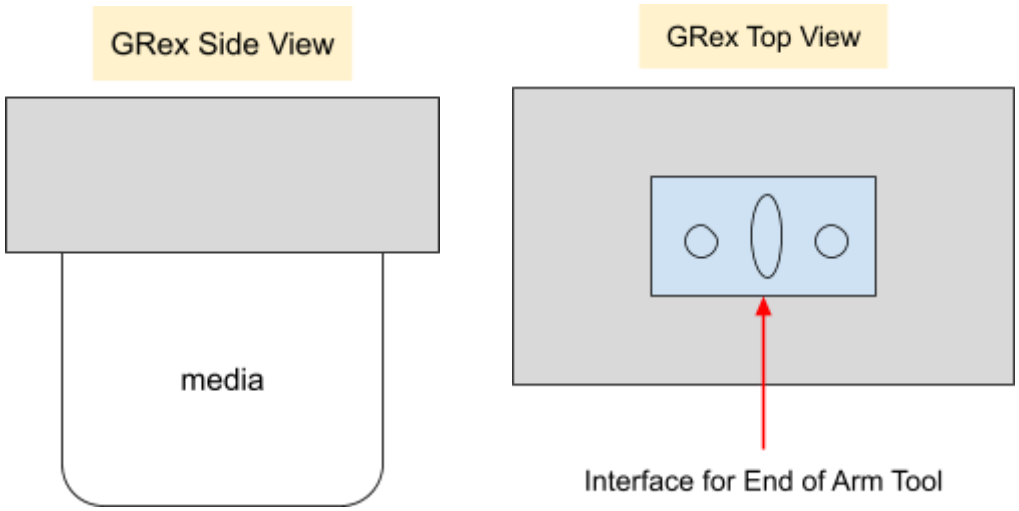
About the Robot:

This 6DOF robot has the following capabilities:

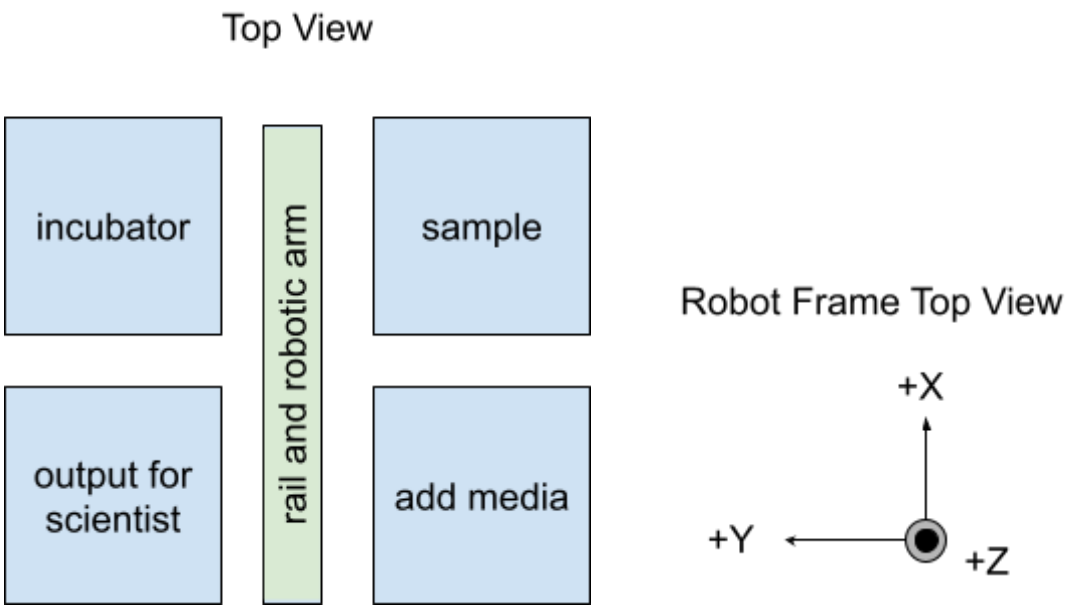
- Obtaining joint positions
- Obtaining Base Frame Cartesian positions
- Obtaining force readings from the base of the robot
- Obtaining velocity readings (joint and cartesian)
- Obtaining acceleration readings (joint and cartesian)
- Moving to specified positions
- Setting speed and velocity (joint and cartesian)

About the GRex Movement:

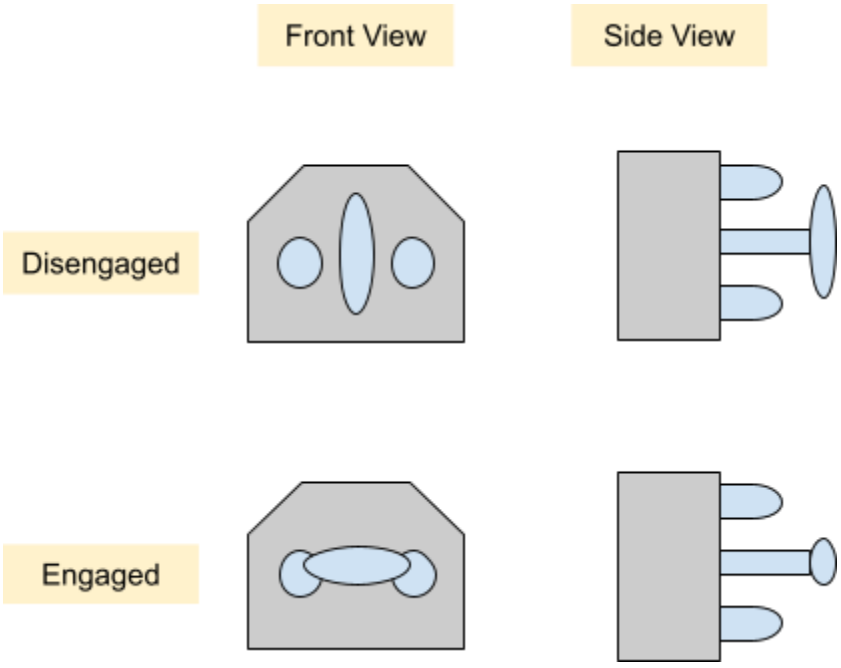
Here at Multiply Labs we grow cells using a GRex that is modified for robotic use. A GRex is a container that holds cells and media. See the diagram below.



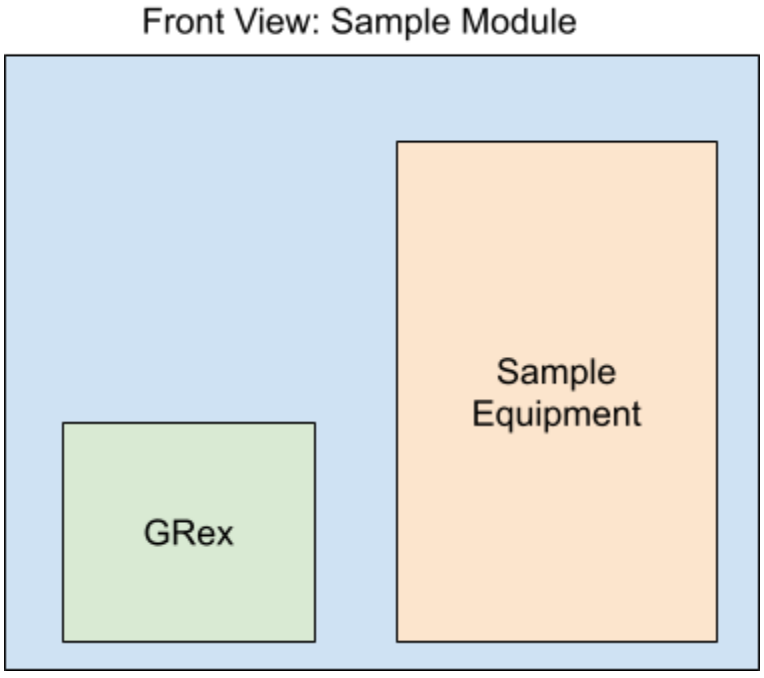
The GRex is moved to different locations within the robotic cluster. Each location within the robotic cluster is called a module. Each module may exist at a different rail position. The robot moves from one module to another using the rail. See the cluster diagram below that shows all of the modules the GRex needs to be in.



The robotic arm has a specific end of arm tool that is able to hold onto the robotic interface of the GRex. To grab onto the GRex, the end of arm tool moves from the disengaged position to the engaged position. This allows the end of arm tool to “lock” onto the GRex cartridge and pick it up. See image below diagramming the different states of the EOAT, and see the image above of the robotic GRex.



Finally, each module has other pieces of hardware in it that the GRex needs to avoid when being picked or placed from the module. The location of this hardware is fixed and will not change. See the diagram below that shows a module with additional hardware in it.



A video of the GRex interaction explained above is shared with you in this zip folder.