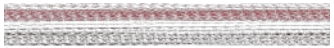
HIPSTER

Autonomous **R**eaming for **T**otal **H**ip **R**eplacement **(ARTHuR)**

Progress Review - 10

Team C: Kaushik Balasundar, Parker Hill, Anthony Kyu, Sundaram Seivur, Gunjan Sethi

November 2, 2022



Goals & Tests

Goals:

- End effector and electrical subsystem completed
- Task prioritization controller tested on real arm
- Watchdog & UI initially integrated

Tests:

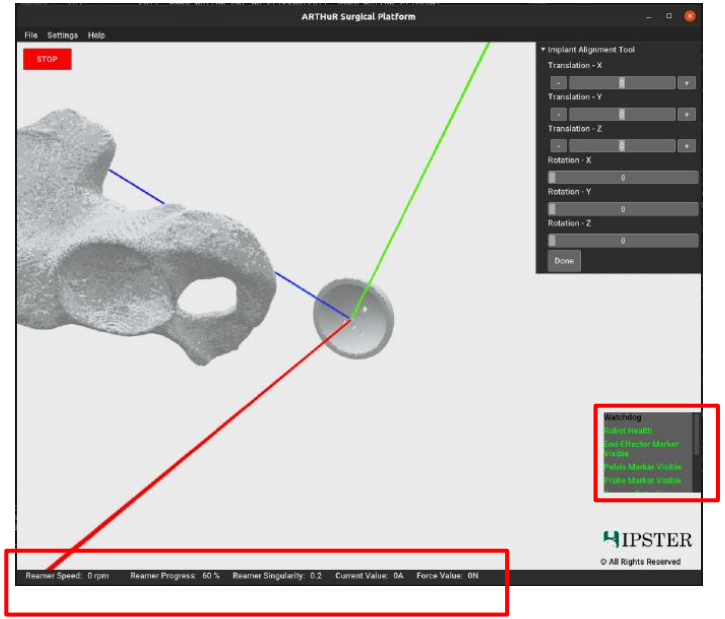
- Test 2: Velocity Controller Tracking Moving Frame
- Test 6: Communicate Cup Implant Pose via UI
- Test 7: Singularity Damping
- Test 8: Joint Limit Avoidance
- Test 9: Task Prioritization
- Test 10: WatchDog+UI
- Test 11: End-Effector Controls - ROS Integration

Progress and Challenges

UI Subsystem (Development and Integration)

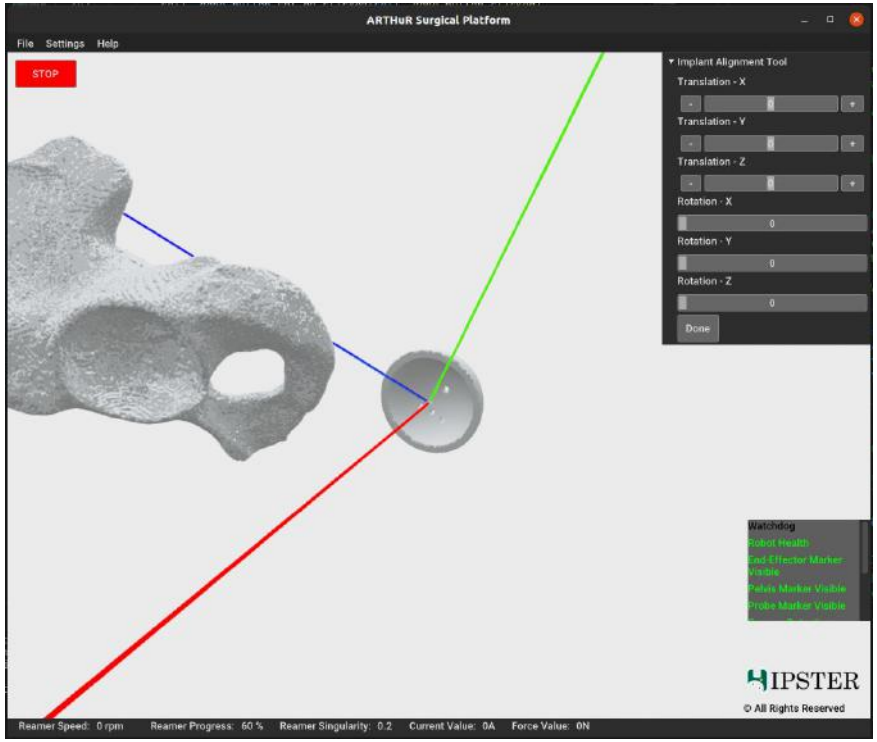
Overall Status	Subsystem	Tasks	Development	Integration	Debugging	Tests
Ongoing	Perception and Sensing	Perform and visualize pointcloud collection	Complete	Complete	Ongoing	
		Select landmark correspondences between pelvis scan and collected pointcloud	Complete	Complete	Ongoing	
		Perform registration and obtain transformation matrix as a result	Ongoing	TODO	Ongoing	
TODO	Controls	Align pelvis with implant and obtain transformation matrix	Ongoing	TODO	TODO	Test 6 (PR10)
		Publish reaming endpoint	TODO	TODO	TODO	
Ongoing	Watchdog	Display metrics	Complete	Complete	Ongoing	Test 10 (PR10)
		Display camera/robot health	Complete	TODO	Ongoing	

Progress



[redacted]

[redacted]


ARTHuR Surgical Platform
File
Settings
Help
STOP
Implant Alignment Tool
Translation - X
Translation - Y
Translation - Z
Rotation - X
Rotation - Y
Rotation - Z
Done
Watchdog
Robot Health
End-Effector Marker Visible
Pelvis Marker Visible
Probe Marker Visible
HIPSTER
© All Rights Reserved
Reamer Speed: 0 rpm
Reamer Progress: 60 %
Reamer Singularity: 0.2
Current Value: 0A
Force Value: 0N

UI Subsystem (Challenges)

- Building UI is hard!
- Currently unable to properly handle all event callback threads; leads to latency issues; UI pauses
- Large codebase and debugging sometimes takes hours
- No elegant way of integrating with ROS
  - Subscriber callbacks interfere with UI callbacks
- Should have started with implant alignment and perfected it
  - Not too late though!
- Fallback plan: only have well-test implant alignment tool + watchdog display

Issue
Mouse middle button causes UI to crash
Inexistant points selected during landmark selection
Unable to get index of selected landmark from pointcloud
Move mouse to update metrics/collected pointcloud – updates only when mouse events are triggered
segfaults during pointcloud collection, handle number of points collected
collected pointclouds don't appear in the correct position

Test 11: End-Effector Controls

Objective	
Verify that the fully-manufactured end-effector is capable of receiving a command to start reaming to a specified end-point, reams to that end-point while maintaining a consistent RPM and not exceeding force thresholds, and reports important values to a ROS topic throughout the procedure	
Equipment	Desktop workstation, robot arm, end-effector, electrical subsystem
Elements	Entire hardware subsystem
Personnel	2 people necessary, one person at the workstation to observe the data being received by certain ROS topics, and one person to observe the arm
Location	NSH B512
Procedure	
1. Verify that the end-effector is connected firmly to the Kinova Gen-3 arm, and that all wires connecting the electrical subsystem to the end-effector are properly connected 2. Following the typical procedure for the fall validation demonstration, set up the arm to track the pelvis dynamically and ream to a specified end-point 3. Click to begin reaming on the user interface and verify that the ballscrew motor begins actuating 4. Once the reaming head makes contact, verify that an axial force is reported in the user interface 5. After contact is made, verify that the reaming motor turns on and maintains a consistent rpm 6. Verify that the reaming operation is not impeded when the arm dynamically compensates for motion 7. Using the stop built into the user interface, verify that the reaming motor and ballscrew motor both stop actuating as soon as the stop is pressed 8. Restarting the procedure from the beginning, verify the stability of the end-effector as the reamer head moves along the axis of the pelvis and that the force threshold is not exceeded 9. Verify that the end-effector reams to the endpoint and the resulting pelvis matches the surgical plan	
Validation	
1. Reaming motor is capable of being turned on and off by ROS autonomously 2. Ballscrew motor is capable of being turned on and off by ROS autonomously 3. Reamer velocity can be monitored via the user interface and remains controlled to a set velocity 4. The axial force applied to the pelvis can be monitored via the user interface and does not exceed the set force threshold 5. Motors stop in the end-effector in less than 500 ms from when a stop command is sent 6. Dynamic compensation does not effect the end-effectors ability to ream the pelvis	

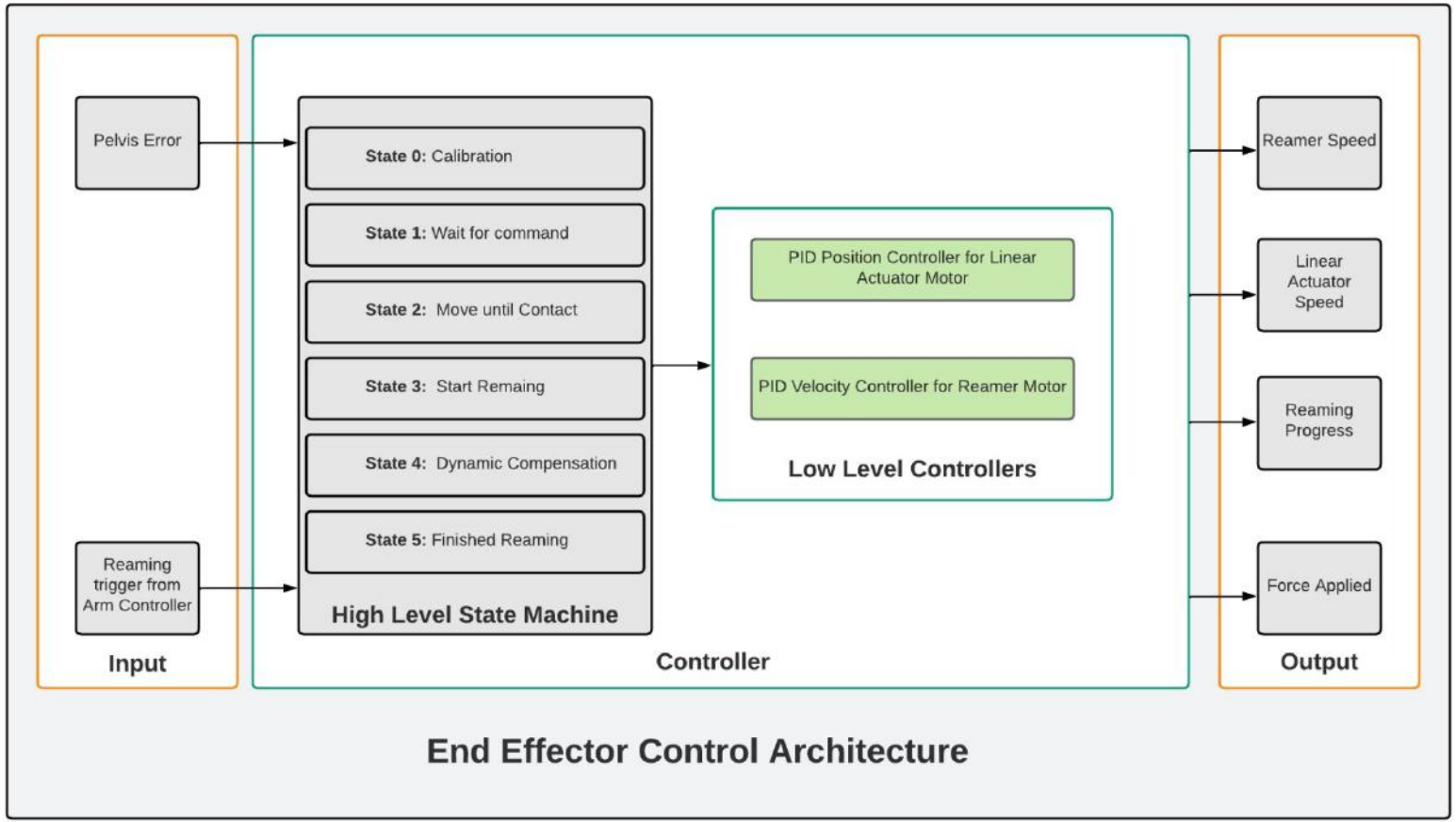
Completed:

- Motors can both be turned on and off
- Velocity can be monitored in UI
- Current sensor calibrated
- Current can be monitored in UI
- Limit switches can limit motion of system
- State machine developed

To Do:

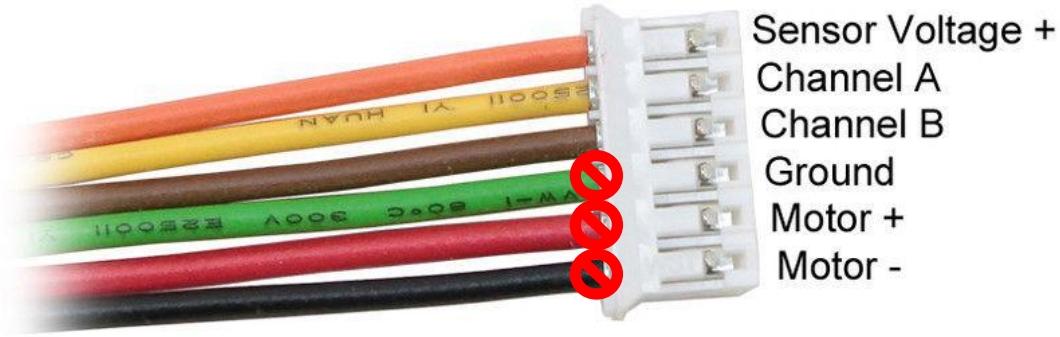
- Correlate current to force
- Finalize state machine controls and integrate with full system

End Effector Controls Architecture

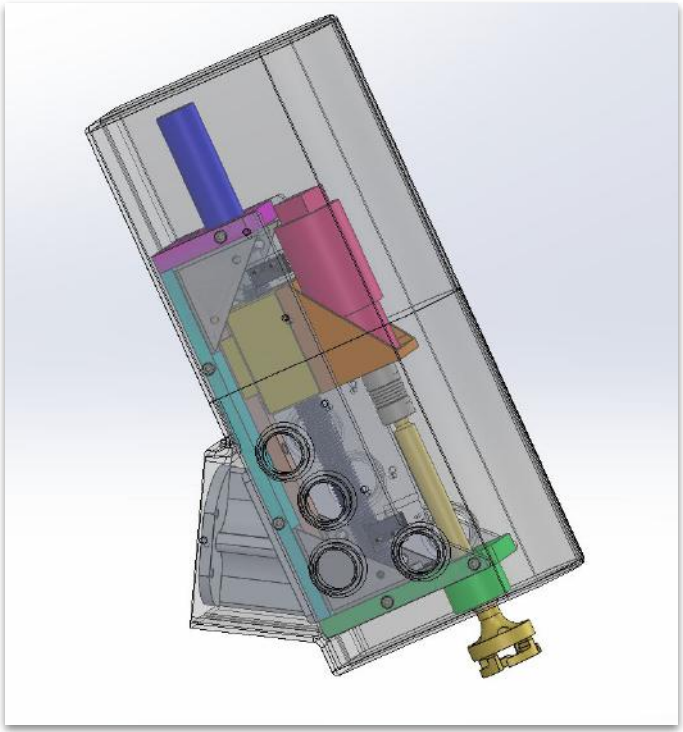


End-Effector Controls Challenges

- Messy code
  - Our current code is based on the previous PID Velocity control we developed last semester, which worked well for what we needed then but has made our current code bulky and inefficient
  - Need to refactor
- Reaming motor recently had the wires pulled off of it
  - Required some janky soldering to the bottom of a hard to reach PCB
  - Ordered new replacement motors and have to implement into system



End-Effector Updates



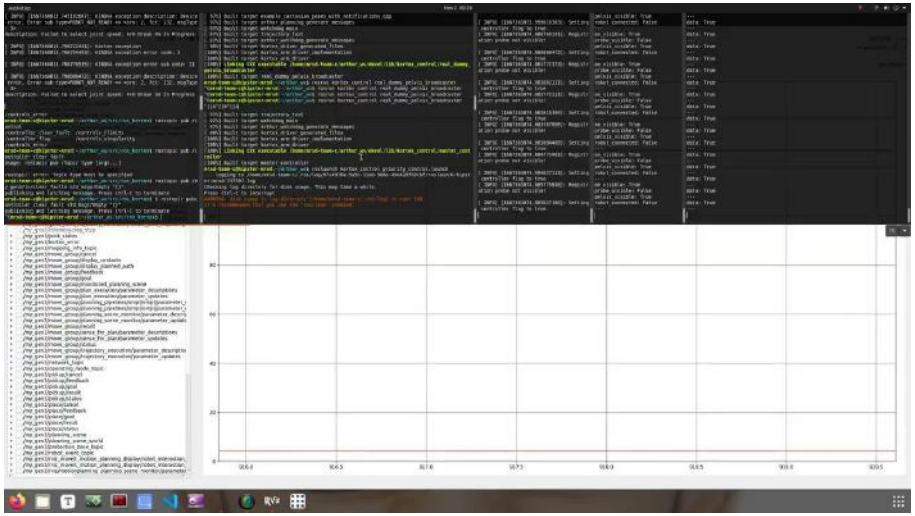
Completed:

- Finalized manufactured end-effector design and began printing it
- Ordered parts off Xometry

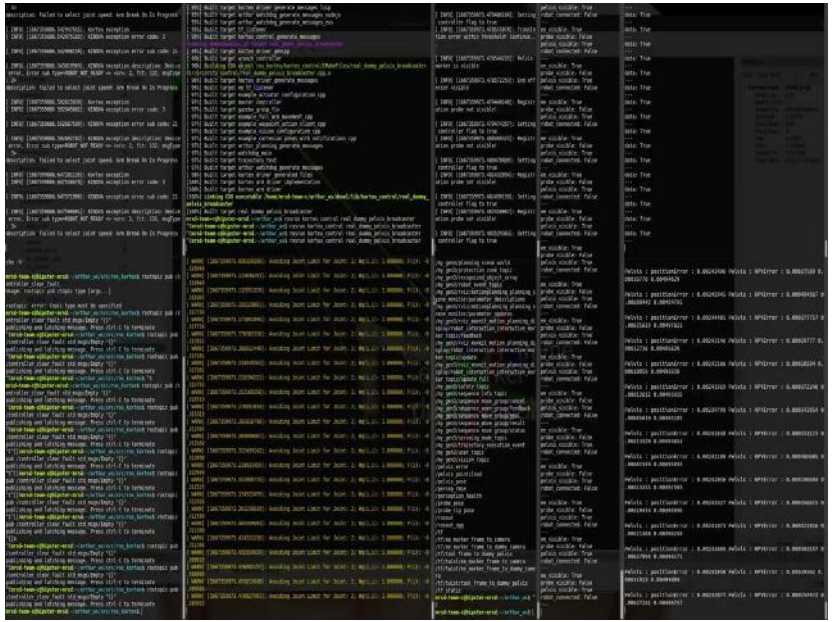
To Do:

- Change reamer handle geometry
- Redesign cover + marker geometries

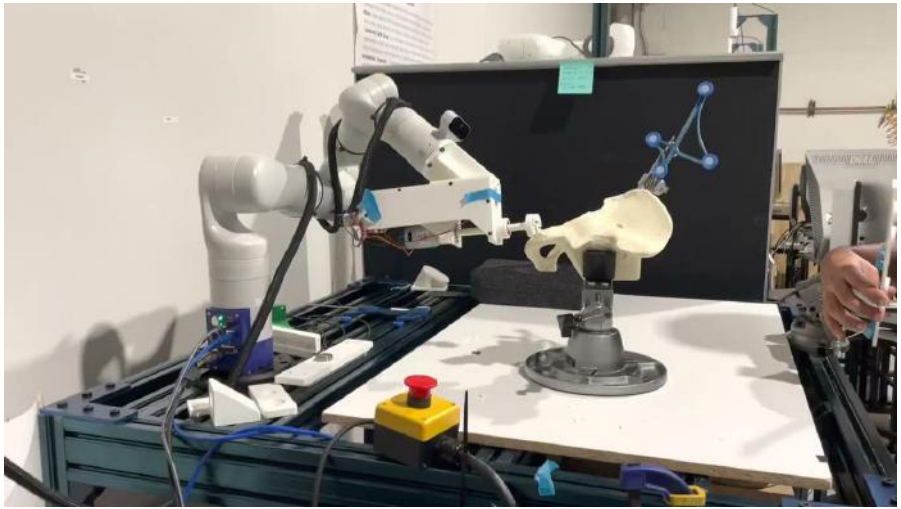
Controls: Singularity Damping on Real Arm

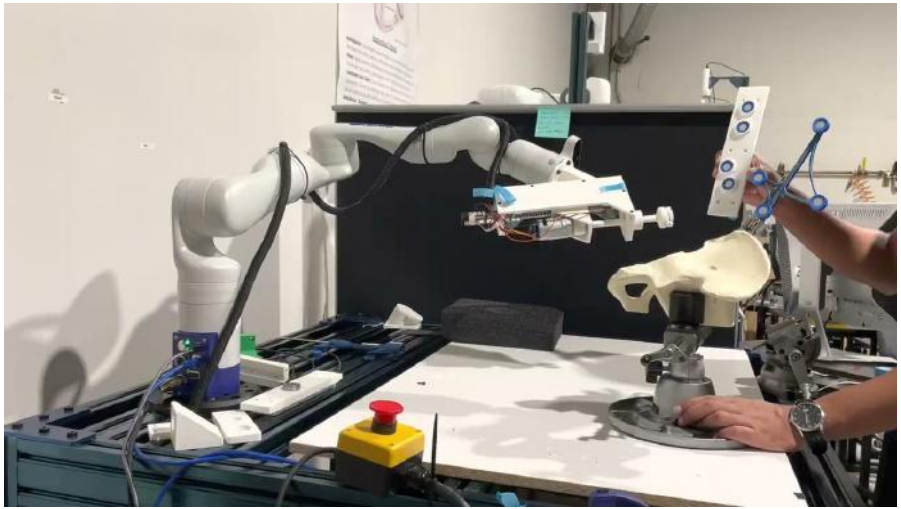


Controls: Joint Limit Avoidance on Real Arm

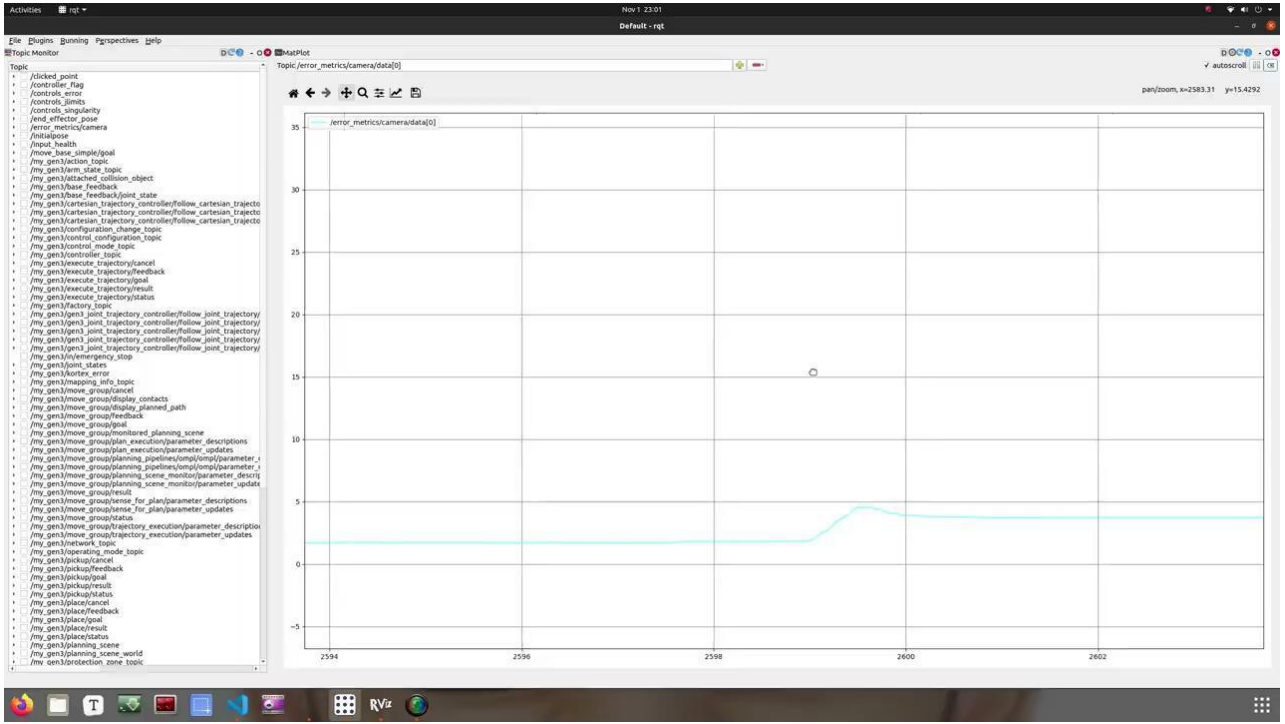


Controls: Collision Detection

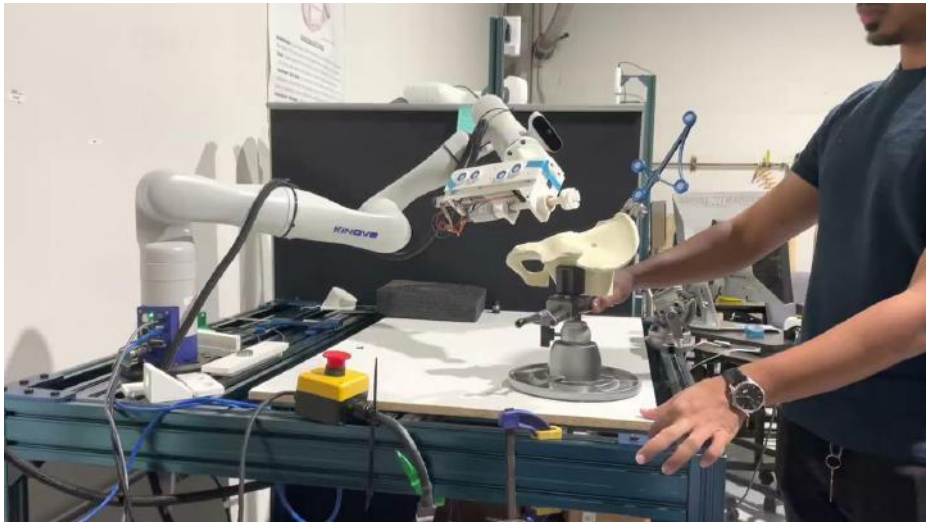


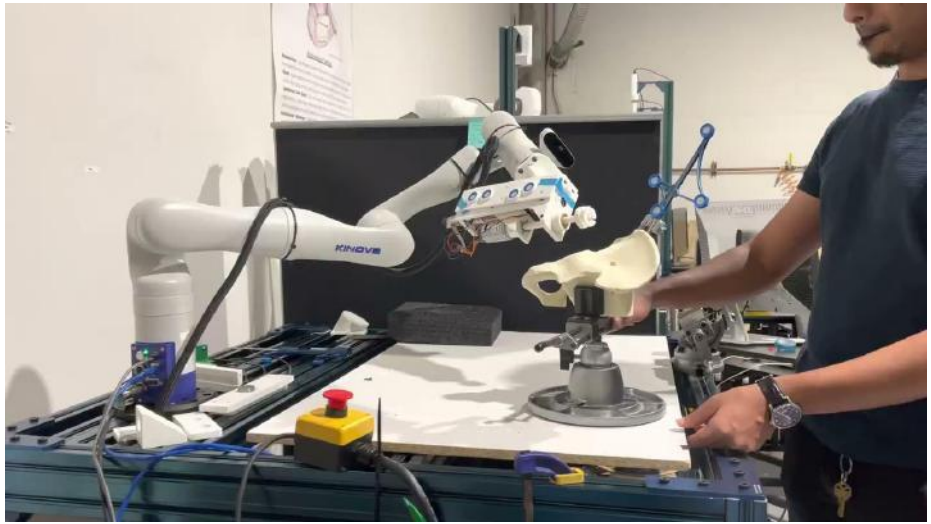


Controls: Task Prioritization on Real Arm



Watchdog + Controls





Future Work

Future Work

- UI Development: Implant alignment/Controls -> Watchdog -> Perception
- End-Effector: Finalize controls and integration with ROS, assemble final manufactured end-effector
- Controls: Integrate End-Effector Controls with Arm Controls and Online Calibration
- RECOVER (it’s been a long two weeks)

Thank you!