PyDM Robotic Arm GUI

James Chen[1], Dehong Zhang[2+]

[1].LCLS Summer Intern, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA 92093, USA.

[2]Stanford Synchrotron Radiation Lightsource, SLAC National Accelerator Laboratory, 2575 Sand Hill Road, Menlo Park, CA 94025, USA.

[+]Contact: dhzhang@slac.stanford.edu

Introduction

The goal was to develop a GUI for a Meca500 robot arm, creating an intuitive interface that is approachable for new users, as well as provide the necessary capabilities needed for complex tasks. With the necessity of remote access and plans of integration into larger systems, the GUI was created using a SLAC developed framework called the "Python Display Manager (PyDM)." In addition to the task of creating a GUI, the project also, in part, tests the capabilities of PyDM and whether it can replace the more commonly used EDM for EPICS based GUIs.

The Robotic Arm

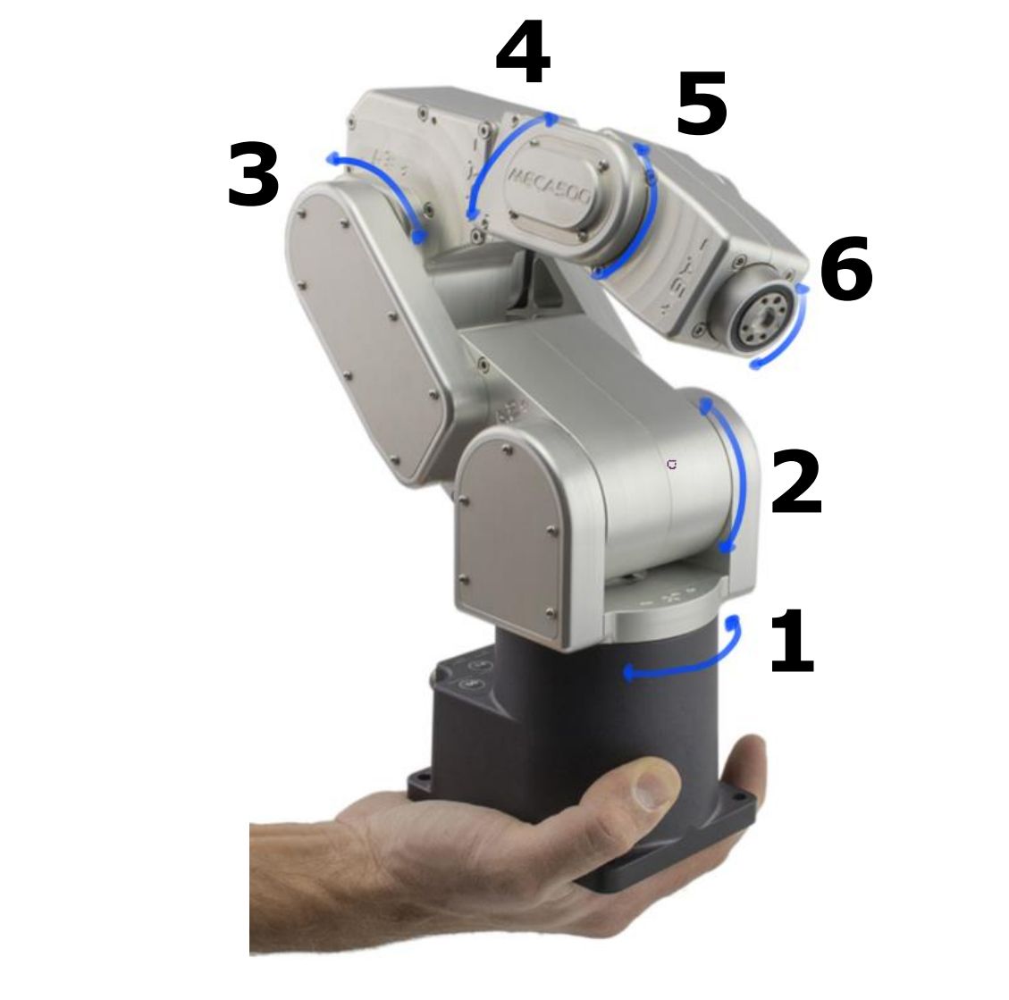


Figure 1 A photo of the robot from Mecademic's official site, providing a sense of scale. The added blue arrows delineate the positions of the six joints and number associated with each joint (where the robot can rotate).

The Meca500 is a six-axis industrial robotic arm that has six degrees of freedom (one for each joint), weighs 4.5 kg, is rated to carry 0.5 kg (up to 1 kg), and can repeat movements with a precision of 5 μm.

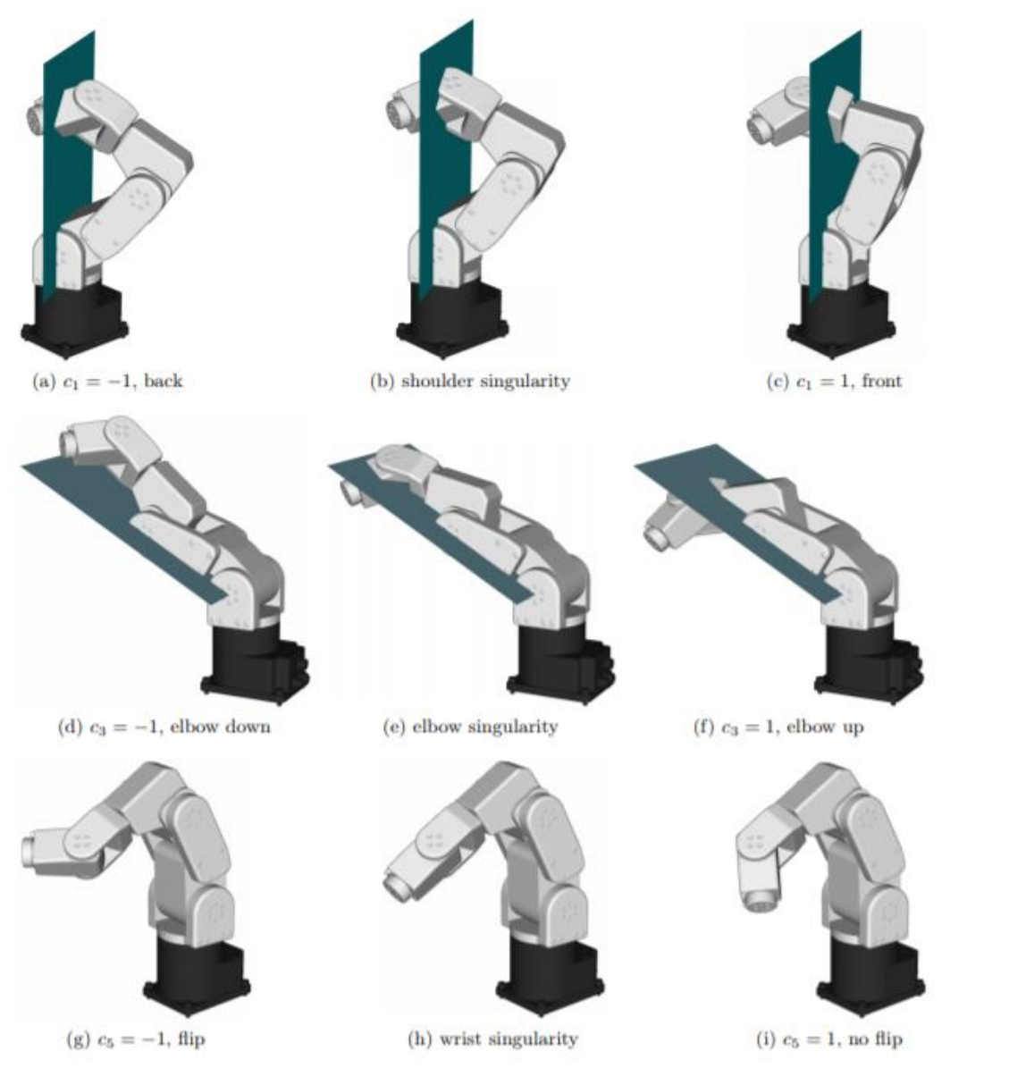


Figure 2 Diagram from the manual displaying the robot's position classified by the variables (c1, c2, c3) and the singularities that exist when the robot is in a position in between them.

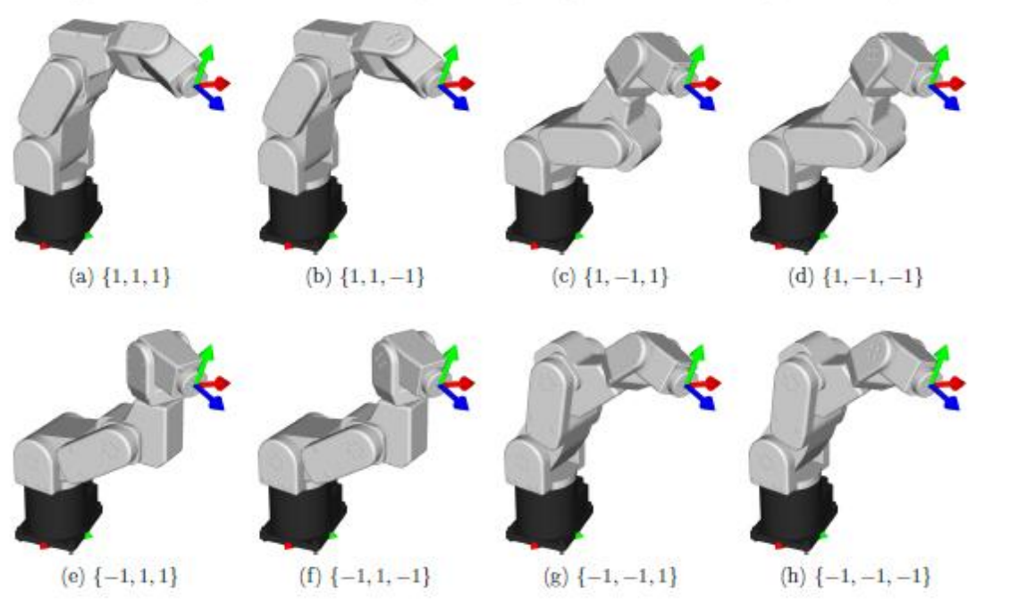
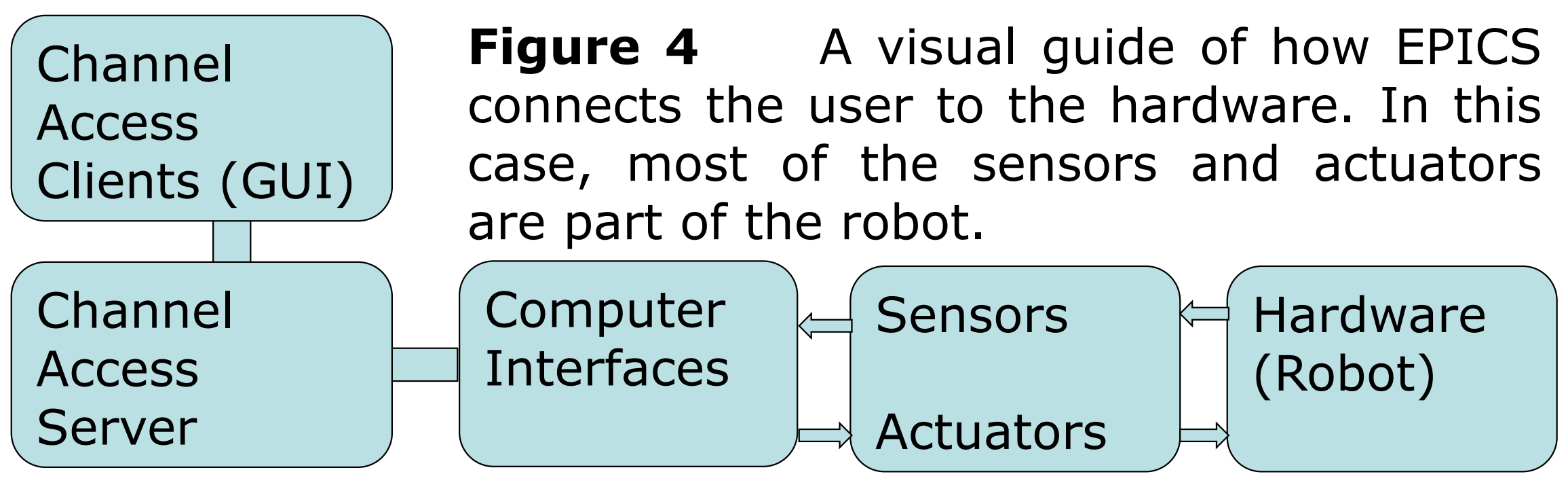


Figure 3 Diagram showing how the robot's end effector can be in same position despite having its joints in 8 different orientations or configurations.

A major part of the robot is the inverse kinematic configurations that govern how the robot moves. Because the robot can reach the same end effector (tool) position with many different joint positions, the robot cannot solve for a single solution without the specification of a configuration. Therefore, the common problem of singularities arises when the robot is in between two different configurations, which causes it to lose 1 or more of its degrees of freedom. When controlling the robot using cartesian commands, the robot's firmware will find an infinite number of possible paths when attempting to pass through a singularity, causing the robot to move unpredictably when near or being completely stuck when passing through a singularity.

EPICS

EPICS is the control system used at SLAC to connect hardware to the various computers on the network. Through this system, it is possible to remotely access and monitor on-site hardware. EPICS was used to connect the robot to the GUI.


Channel Access Clients (GUI)

Channel Access Server

Computer Interfaces

Sensors

Actuators

Hardware (Robot)

Figure 4 A visual guide of how EPICS connects the user to the hardware. In this case, most of the sensors and actuators are part of the robot.

PyDM

The Python Display Manager is a wrapper for PyQt, a Python-based GUI development software. PyQt's Qt Designer enables the user to drag-and-drop channel accessible widgets on a screen as well as the ability to make use of Python's scientific repertoire. This allows the user to create more complex interactions between the different elements of the GUI as well as enable the ability for real-time data analysis.

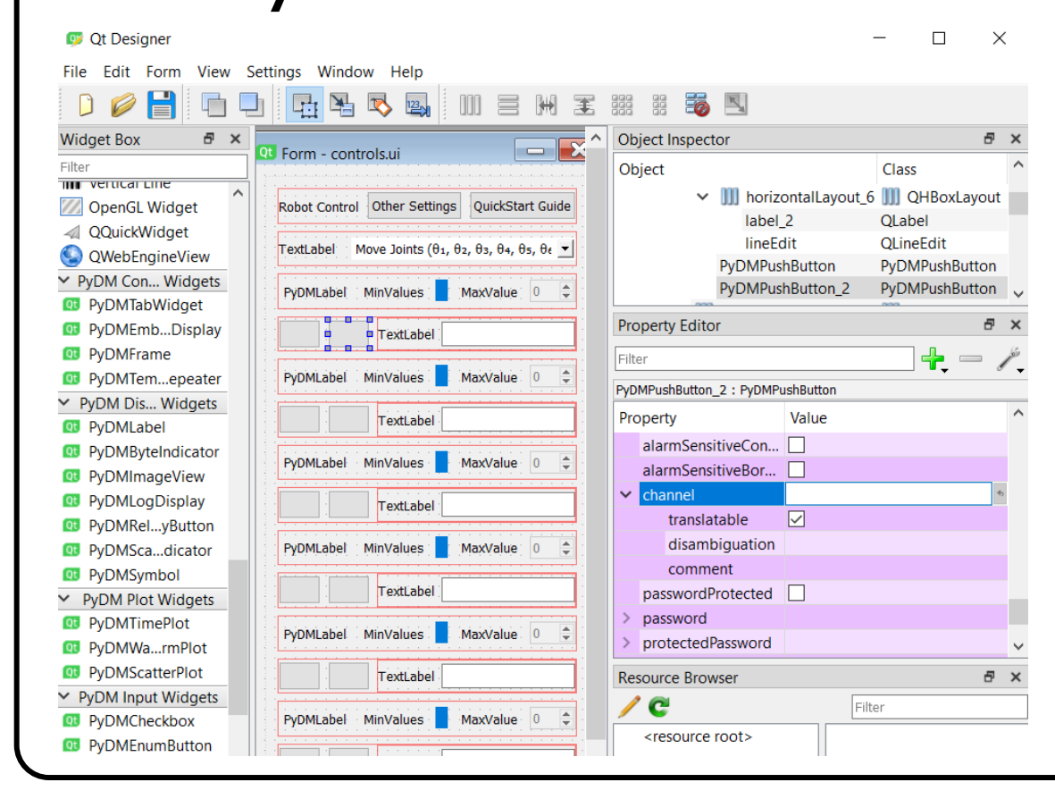


Figure 5 A screenshot of a prototype PyDM GUI, showing a button that can connect directly to a channel. Macros can also be used in the "channel" section to use the same GUI for the same type of hardware.

The GUI: The Basics

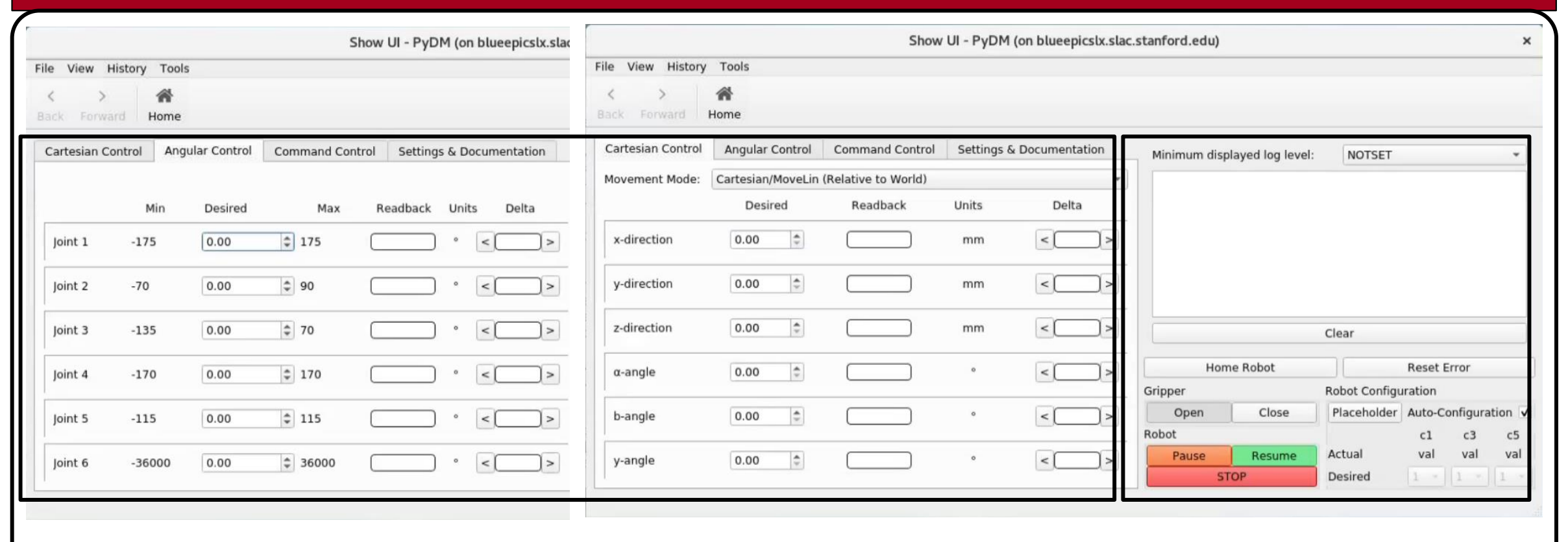


Figure 6 A showcase of the Angular and Cartesian tabs

These tabs provide users with a brief overview of how the robot works and allows the user to issue simple commands one at the time like jogging the joints or providing a set of coordinates for the robot's tool.

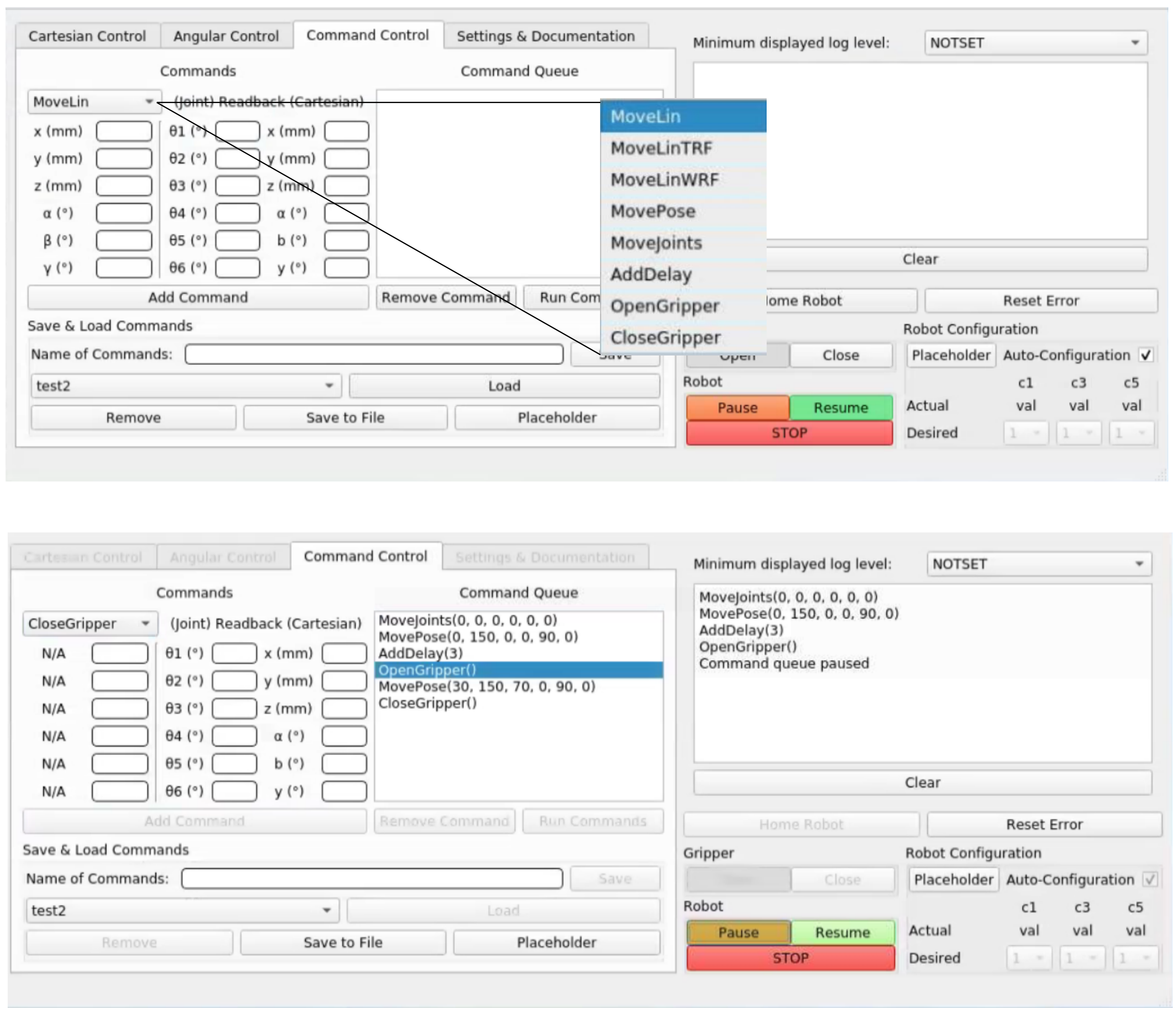
The right-side panel displays commonly used features such as the message log, where the robot's feedback and error messages will be sent. The bottom left corner allows the user to control the gripper. The bottom right allows the user to set the desired inverse kinematic configurations.

The "Settings and Documentation" tab provides links to documentation on how to use the GUI as well as robot settings like limiting the robot's movement range, setting physical limits (joint velocity, linear acceleration, etc.), and changing the reference frames of the robot base and end effector.

The GUI: Command Queue

The "Command Queue" tab allows the users to queue up all possible commands and adds them to the list widget as Python strings. From there, the individual commands can be removed, or entire command sets can be saved to a Python dictionary as a variable in the program. For more permanent commands, the commands can be saved to a JSON file via the "Save to File" button. These commands will be loaded into the drop-down menu at the start-up of the GUI.

Upon clicking the "Run Commands" button, the program will loop through the commands in the command queue, split the strings up and send the individual sections to their associated connections. The "Run Commands" button is multithreaded so that the "Pause," "Resume," and "STOP" continue to function, while the other buttons that can interfere with the process are disabled. Most of these processes will send a message to the log to signal errors or highlight the completion of an event.



Minimum displayed log level: NOTSET

test2

Command Queue:

```
MoveJoints(0, 0, 0, 0, 0, 0)
MovePose(0, 150, 0, 0, 90, 0)
AddDelay(3)
OpenGripper()
MovePose(30, 150, 70, 0, 90, 0)
CloseGripper()
```

```
MoveJoints(0, 0, 0, 0, 0, 0)
MovePose(0, 150, 0, 0, 90, 0)
AddDelay(3)
OpenGripper()
Command queue paused
```

Figure 7 Showcasing the command queue tab and an example of when the commands are running.

Conclusion

Although the documentation for PyDM is currently sparse and there is a learning curve to PyQt, PyDM has vast benefits of allowing users to develop simple GUIs quickly (drag–and-drop) and gives users the option to develop more complex tools. Even though the Meca500 robot does not have a focus in data collection, PyDM allows the development of data collection tools with real time feedback, which can be extremely useful.

Acknowledgments

Use of the Linac Coherent Light Source (LCLS), SLAC National Accelerator Laboratory, is supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under Contract No. DE-AC02-76SF00515.

Date: 09/10/2020