ENAE380 FINAL PROJECT PROPOSAL: JR3 Force-Torque Sensor Visualization Program

Romeo Perlstein

Section 0102

Like most of my aerospace engineering colleagues, I'm incredibly passionate about space. One of the most incredible things about aerospace engineering is the large span of disciplines it encompasses, from ground support systems to interstellar spacecraft, the possibilities seem almost endless. My experiences at UMd have led me to be passionate about imbedded systems, extreme robotics, and human-robot interactions in space. However, I find it very important to utilize the benefit of aerospace being such a widespread field of interest to try and learn as much as possible at more than just what I'm passionate about. I enjoy learning about how a system works, rather than just focusing on one aspect. To support my passions, I'm hoping to work with NASA in the NEXIS department at the Goddard Space Flight Center.

For the ENAE380 final project, I would be interested in writing a ROS 2 – based software stack for reading in data from JR3 force-torque sensors. The program would have two main functionalities: A simulated aspect that allows for debugging and testing in the absence of physical hardware, and the ability to read and display data from either the simulated or real data. The display of data would utilize RVIZ and the Unity game engine; the Unity game engine has a built-in physics engine, allowing for "real" forces and torques to be measured by a simulated JR3 sensor. The goal is to incorporate ROS 2's ros2_control functionality, such as its *HardwareInterface::SensorInterface* class, as well as to establish Unity functionality for future work. The hope is to have a program entirely separate from any existing lab software, that can be easily integrated after the final product is completed.

The next few steps would be as follows: First, I need to establish a good connection with the physical JR3 sensors. This requires system-level interaction, such as installing and building custom drivers, and successfully communicating with the sensors. The next step (or a step parallel to the first) would be to begin developing the groundwork for the program. This would involve writing any C++ or Python -based classes or nodes. After that, the final integration steps would occur. This would involve incorporating ros2_control into the main software, as well as creating the ROS-Unity connection.

To accomplish this, I will need to work on my Linux terminal navigation skills and commanding Linux based operating systems, so that I can establish contact with the physical sensors. I will also need to begin researching ros2_control sensor functionality, ROS-Unity connection strategies, and sensor simulation techniques using Python and C++, so that I can incorporate all of my desired outcomes into the program. The hope is that this project will allow me to gain knowledge on ros2_control, communicating with hardware and sensors, and developing simulations using basic languages like C++ and Python. Also, this program would benefit the Space Systems Laboratory in the long term, as the JR3 force-torque sensors are the main sensors used by multiple systems in the laboratory.