Team Name: Project 3 33
Team Members: Alex Perez, Daniel Shmul, Charles Richardson
Project Title: Subjugator Navigator (of data)

Problem: Robot Operating System (ROS - a framework for writing robot software) bags are recordings of live or simulated data that can be used to analyze the behavior of robots or autonomous machines. The Machine Intelligence Lab (MIL) at the University of Florida heavily deals with ROS bags so as to analyze data for their autonomous submarine. However, it seems ROS is limited on the statistical analysis methods that it offers, which can make it hard to analyze the recordings.

Motivation: Sometimes these ROS bags are filled with hundreds of thousands of lines of data that can be difficult to analyze and digest. In fact, just five seconds can record up to 1000 data points. There are some tools that help us graph desired values, however, most involve having to replay the recording, which can get tedious or time consuming. Additionally, if you choose to do any other statistics on the data, you must parse through the raw data yourself. We want to make this whole process easier.

Features: We want to create a preliminary tool that takes a ROS bag and allows for various statistical options geared towards helping MIL members analyze their robot's data which will speed up the development, debugging, and analysis curve. The problem will have been solved when we can extract the following from bags of data:

- Max velocity
- Max angular velocity
- Range of time for near constant velocity
- Range of time for near constant acceleration
- Mapped forces

Data: We will be using data that will be generated from the MIL submarine's simulation. Given a ten minute simulation, the simulation will produce the required amount of data (in the form of floats) that will contain:

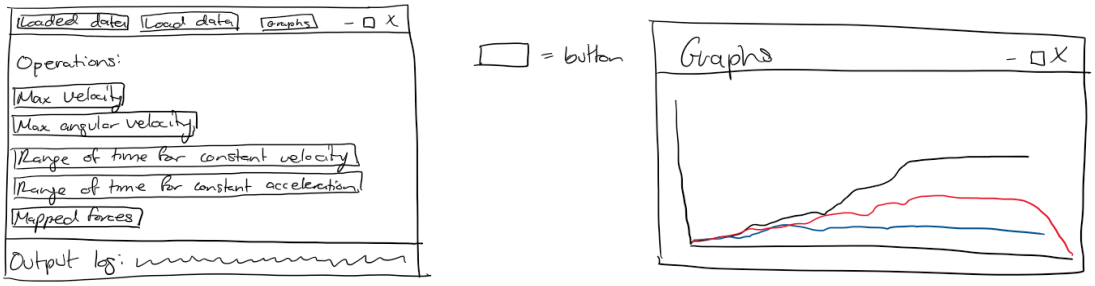
1. Odometry data
    a. Position (x, y, z)
    b. Orientation (x, y, z, w)
    c. Linear velocity (x, y, z)
    d. Angular velocity (x, y, z)
2. Wrench data
    a. Force (x, y, z)
    b. Torque (x, y, z)
3. Imu data
    a. Orientation (x, y, z, w)
    b. Angular velocity (x, y, z)
    c. Linear acceleration (x, y, z)

For the purposes of this project, we will focus on linear velocity, angular velocity, linear acceleration, force, and torque.

Tools: ROS is used to help obtain the data set (the simulation was built within ROS), which could be shown in the video if we wanted to. However, once we get the csv data set, everything would be done in C++ and through a CLI. We are also hoping to use a GUI for the visual to provide menu options and display results.

Visuals:
Layout of how the menu system would work with a corresponding submenu for each option.

1. Load data
2. Find data at time *(user would be given an allowed time range and allowed to choose a time)*
3. Find all nodes with velocity greater than x *(user would choose value x)*
4. Find constant velocity *(user would choose x, y, or z)*
5. Find constant acceleration *(user would choose x, y, or z)*
6. Find MAX *(user would be provided a list of data options to pick max of)*
7. Graph for the data *(user would be provided a list of data options to graph)*



Strategy: Some data structures algorithms we may include:

- Trees could be extremely useful in looking for data. Specifically AVL trees. We may have to create our own or adjust the project 1 AVL trees if we can't find a premade data structure that fits what we want (a node that contains all the specified information).
- We could also use it with a normal BST and compare the time required for searching to an AVL tree.
- An unordered map could be useful in attempting to access data given a certain key (such as time). This would be good given the constant access of an unordered map.
- Algorithms we will need include insertion, and searching (level order and binary). More algorithms may appear as time progresses.

Distribution of Responsibility and Roles:

- Frontend (UI) - Daniel and Charlie
- Backend (DB) - Alex

References: Machine Intelligence Lab software will be used in order to obtain a dataset.