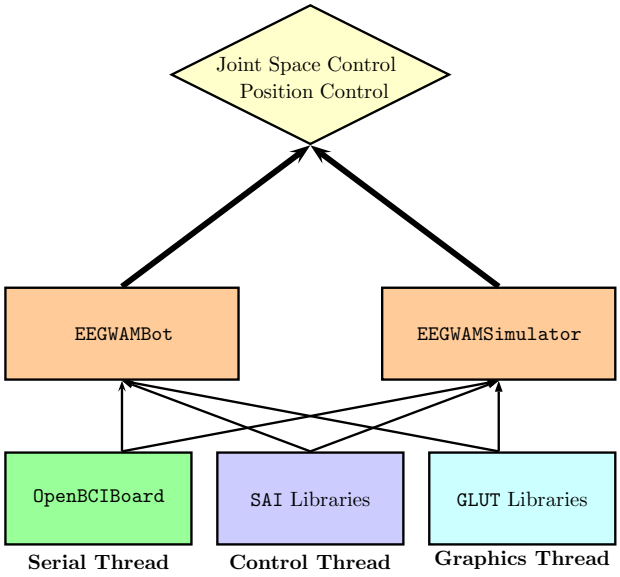

Figure 2: The serial thread, control thread and graphics thread are all necessary to control the robot in real time using the EEG device. Not included in this diagram is the plotting component, which is shown simultaneously using Cython.

Using a `matplotlib` plugin for C++, we integrated Cython into our program to do real-time plotting of the last five seconds of EEG data. This allowed us to improve our debugging capabilities as we reached the homestretch of our project.

2.5 Serial Communication

Because we use the OpenBCI as our central EEG system, we write a serial communication interface in `C++`. To do this, we convert the already-provided Python interface into a `C++` interface to parse data packets streaming from the EEG system into our data processing algorithm for analysis. Because the computer has to handle multiple threads, we used a thread-safe library called `libserial` to read and write to the port. The structure of the data packet is shown in Figure 3. Our implementation of this is contained in the class `OpenBCIBoard`. In order to actually transfer the data analysis onto a robot action, we allow a callback function as an input into our streaming function.

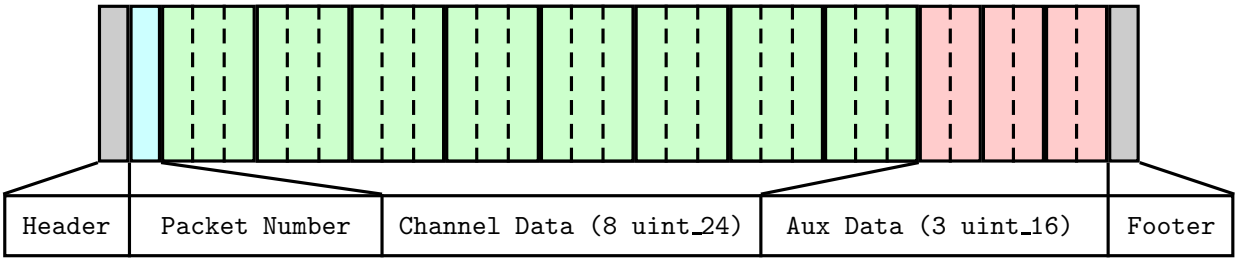

Figure 3: The structure of the data packet, where each block represents a byte of data coming in from the serial port. The auxiliary data represents the accelerometer data which we use for Task 2.