**Weekly Report – W15 Spring 2023**

**Task & Problem**

1. Edit the controller in Arduino board, and do some principal experiments if possible;
2. Derive the dynamics of the whole system according to piecewise constant curvature theory;

**Solution**

1. Controller design

The controller design largely depends on the dynamics itself due to the properties of model based control, it will be discussed in the next section. This part will focus more on the debug issues.

(1). All the issues with device recognition have been fixed

Last week because I have added too many new features to the original code, however, during that period, I haven’t conducted any tests about it, that is the reason why the codes would have so many bugs at the end of last week.

Though the focus of this stage is not to consider so much about the user experience, the bugs have affected the normal utilization of the code, so I have no choice but to fix them all. So now the code has some new features compared to last week, which can be seen as follows,

* The experiment data will be saved in local files on PC only if there are at least two devices connected with the PC and one of them must be the base coordinate IMU; otherwise, it doesn’t make sense to record any data without the base, since the two arms’ angles and angular velocities are computed based on and relative to the base’s, so just connecting the IMUs mounted on the end effectors will cause wrong data, which we don’t need; on the other hand, to realize the control, we must need the data from end effectors of the arms, so just connecting Arduino board (knowing the states of human) cannot satisfy our requirements either.
* The code was originally to be designed to allocate the vacant port in sequence for new devices, for example, when the IMUs for arm 1 and arm 2 end effectors were connected for the first time, we can write them into system struct as arm(1) and arm(2), considering that we could have more sensors in the future, like another two IMUs mounted on the middle (joint) of the two arms, we can define them as arm(3) and arm(4), if for the second trial, the arm(1) and arm(2) IMUs were pulled out, and (3) and (4) were plugged in, they will replace the space which has been allocated to (1) and (2), so the calculations will mess up.

In the new version released, another field called “system.devices.history” was set up to store all the device connection information for the same PC in the past, and it will be “inherited” to the next “generation” (next test on the same PC), when new devices were plugged in, the history list will be updated; if it is found that there exist certain current connected devices are already in the history list, they will be allocated with exactly the same COM port as shown in the list so that the issue demonstrated above would not happen.

* Sending stream data from MATLAB to Arduino board has been added as a new feature as well. Since the IMU mounted on the backpack was taken by Charanjit, currently we cannot read serial data from human using configureCallback function in MATLAB so that the communication between MATLAB and Arduino board part was temporarily written inside the function that can realize the communication between MATLAB and ESP32s.

Until this Friday, I haven’t found out any promising solutions that could verify if the stream data can be successfully sent to Arduino board. One of the reasons is that we cannot send data from MATLAB to Arduino and meanwhile use Arduino IDE serial monitor to check out if the expected output will be showcased there, it will indicate that the port is occupied. Thus the only solution is that we can send the data first and then let the code in Arduino board to send exactly the same data back to MATLAB again, however there are some strict rules for the formatting of the data sent.

After multiple times of tests, it turns out that if we use writeline command in MATLAB, the input must be a scalar string rather than the array which is composed of int or double class numbers, and after dealing with this issue, the data can be successfully received by Arduino. However, here comes another problem, the data sent to Arduino will be changed into pointer char, it needs to be change into string first, then the key states of arms can be picked out and be transferred into int or double again to finish the sequent calculation next. The difficulty is that I haven’t programmed with C/C++ for years, and the rules for coding in Arduino seems like to be a combination of C and C++, lots of the functions were not fully applicable for Arduino, I have to spend some time getting familiar with it.

1. Dynamics of the whole system

State: pending.

One of the assignments for Dr. Wang’s course is to write a chapter for a e-book related to robotics topic, and everyone in class will be responsible for one chapter, I decided to write something about different modelling methods, so I think I can derive the dynamics at the same time.