# **Weekly Report – W13 Spring 2023**

## **Task & Problem**

1. Fix the range (or interval) of the angle from to according to the guidance from Manu;
2. Apply a filter to the angular velocity for both and to eliminate the noise as much as possible;
3. Combine all the previous work, the communication between PC (MATLAB) and IMU (ESP32), and between PC (MATLAB) and Arduino board;
4. Auxiliary parts designed for the backpack;
5. Propose a draft version of a more explicit experiment schedule.

## **Solution**

1. angle interval from to

According to the original definition of the angle provided by Manu, it should be from rather than from , which is due to the limitation of the function . It doesn’t mean that the range from is wrong, we can post process the angular velocities individually, however, this will aggravate the burden of computation, because in the real experiment case, the arm will frequently swing around the boundary and back and forth, causing the velocities quite unstable, and setting a specific algorithm will slow down the data reading process, so it’s better to solve this issue from the source, the angle itself.

However, here brings about another issue about calculating the velocities, especially for angle, because at some points it will change from to 0, my previous algorithm can only deal with the velocities for the angle change from to , or vice versa; now after calibrate the angle interval, we can apply the similar rule as my previous algorithm, to judge the angle variation between the current and former step, if it is greater than 6 rads (it will be too late and not accurate to compare with ) and a simplified version of my algorithm is shown as follows,

if

if

elseif

end

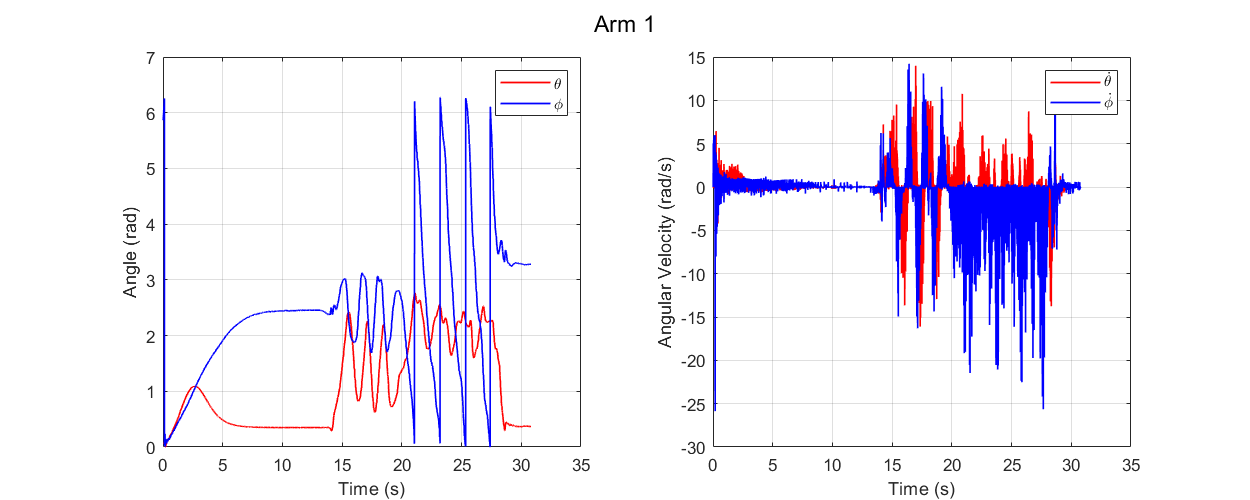
end

where is the sampling rate/frequency.

And the specific effect before and after is shown in the two figures below.



**Fig. W13-1** The real time data collection after adjusting the angle interval but without calibration on its angular velocity



**Fig. W13-2** The real time data collection after adjusting the angle interval with calibration on its angular velocity

We can clearly see that the angular velocity for will drop down to a normal range, and more tests were performed to find if there still exist very large velocity for motions in different directions, it turns out no such issues any more, and our algorithm works.

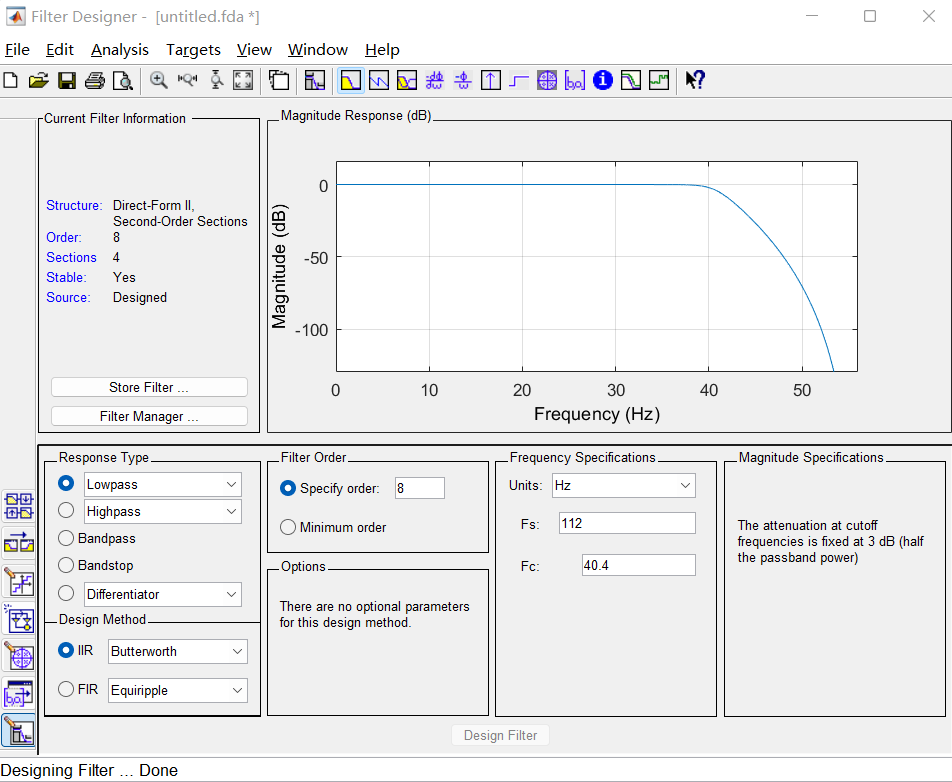
1. Filter design to eliminate the noise for both angle velocities

#### (1). Filter design

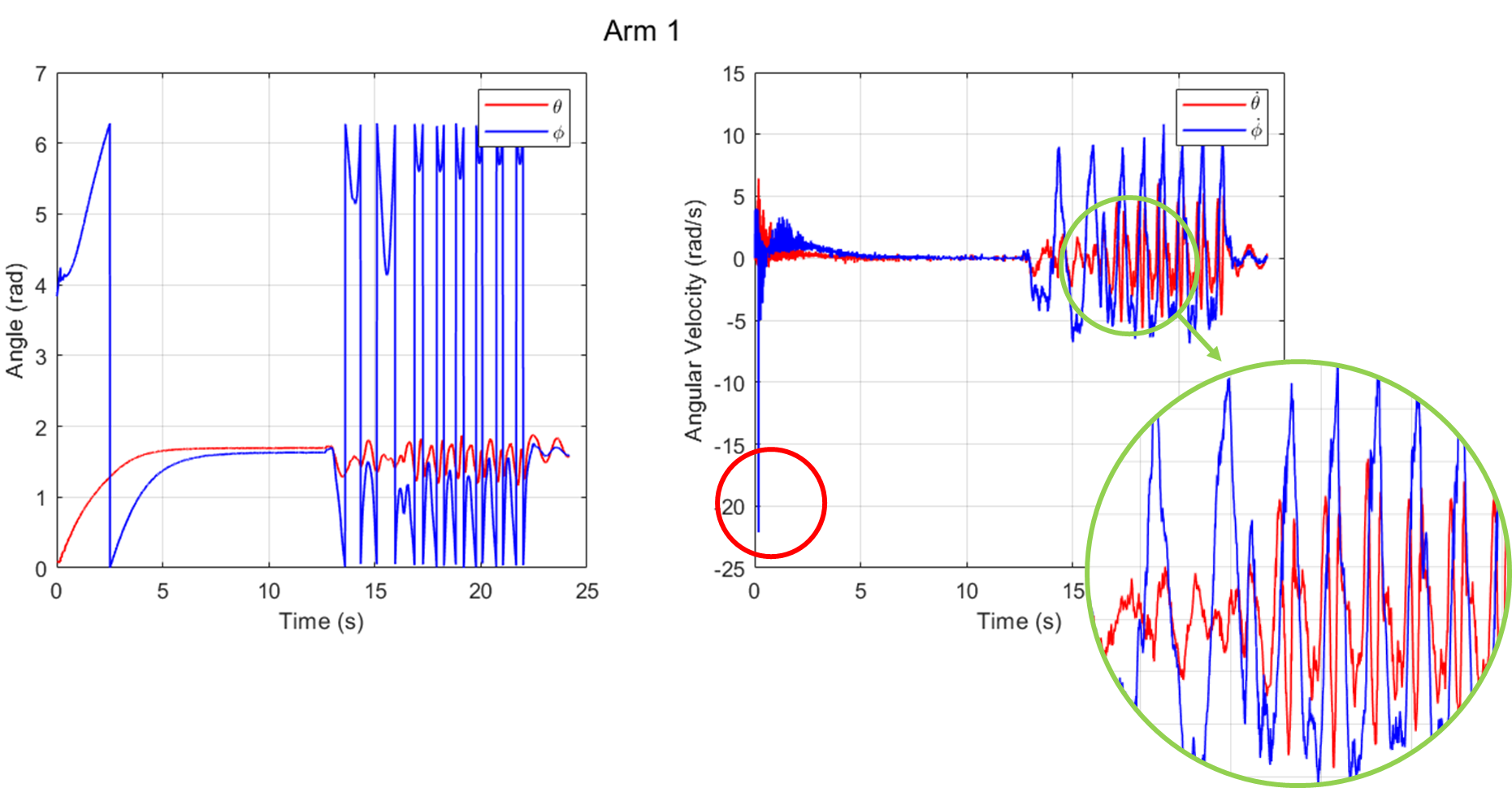
As shown in the Fig. W13-2, though after adjustment and calibration, the angular velocities are correct right now, however, compared with the real data collection from angles themselves, the angular velocities contain too much noise that it will affect the final effect of the controller, to let the controller work properly, we need to give it a more smooth input, in this case we shall design an applicable filter onto the velocities.

Last week I have tried to add a Kalman filter and some other filters we would commonly use, the effect were not so promising. And due to some background knowledge about Kalman filter, it is more applicable for dealing with irregular noise (not the ones we would meet very often like white noise or red noise), it will calculate the current states based on many former steps of data and cause some expected servo lag. Though there exist some simplified version of Kalman filter for practical use, it will still take up a lot of computation resource, that’s reason why I decided to design a much simpler and conventional filter.

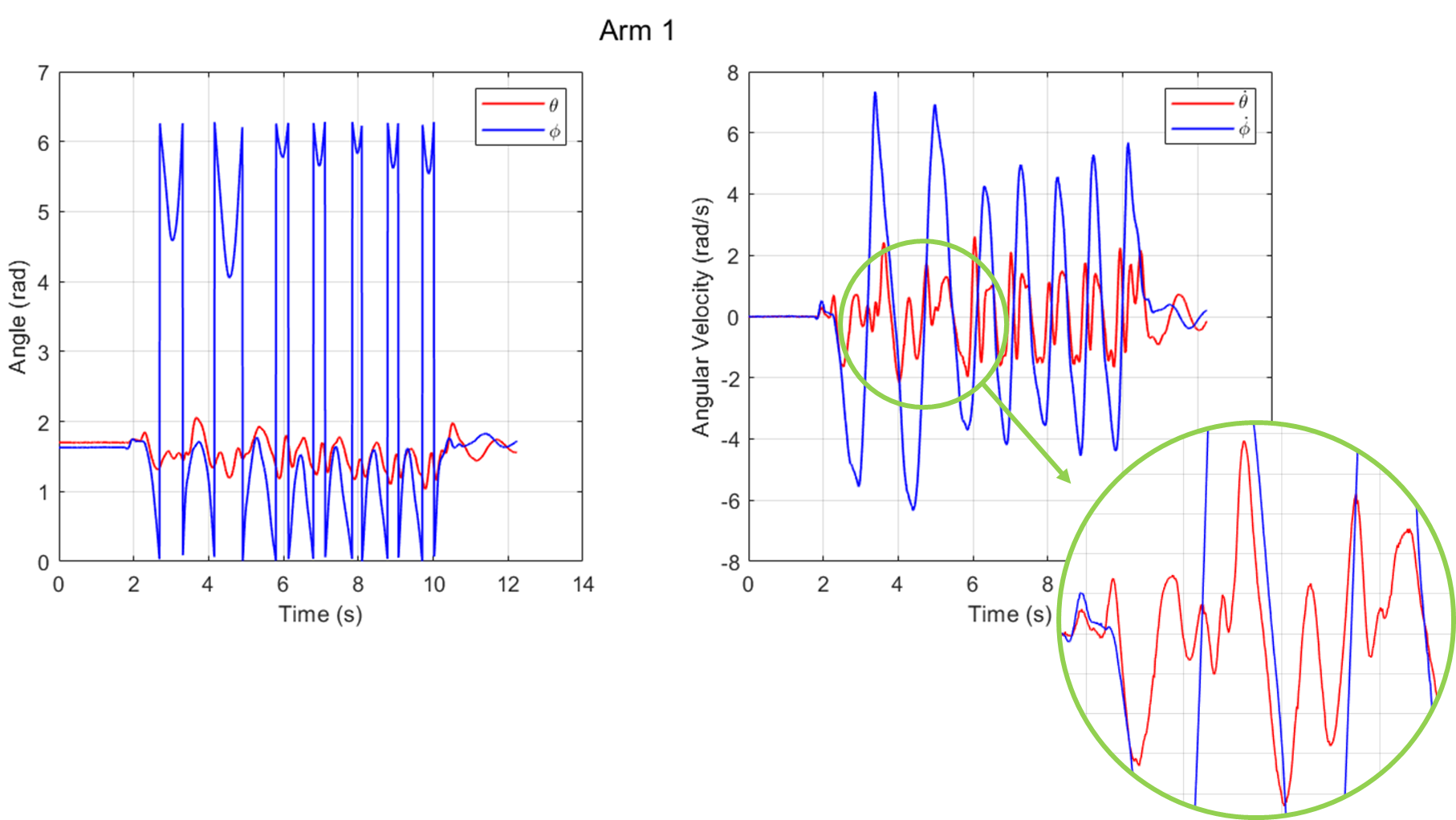
Using the Filter Design Toolbox in MATLAB shown below, I designed a common lowpass filter, with method IIR Butterworth, filter order to be 8, sampling frequency to be 112 Hz. And it is worth note that the proper cut-off frequency for and are slightly different, after hours of tests and comparison, they were determined to be 40.4 Hz and 38.38 respectively. Any value below them will cause serious anamorphose.

**Fig. W13-3** The interface of the Filter Designer in MATLAB

And the comparison between the real time data collection before and after adding a filter is shown as follows.



**Fig. W13-4** The real time data collection without a filter



**Fig. W13-5** The real time data collection with a filter

And I have tested the code for different arm moving speed scenarios, it has a more significant effect for fast moving situations.

#### (2). Test mode design

From what has been highlighted in the red circle in Fig. W13-4, during the initialization process (first 10 s after the power is on), there will be some extreme change about the angle, which will cause a very large velocity at the very beginning. This will distort our plotting scale adjustment and meanwhile the data for the first 10 s are unexpected for our experiment, so I designed two data collection modes: “authentic” and “perfect”.

For the “authentic” mode, no filter will be applied, and the data will be recorded entirely including the first 10 s’ initialization process; for the “perfect” mode, the lowpass filters will be utilized **after** the initialization, because the data in the first 10 s contain too many errors, we don’t anticipate them to affect the execution of the filters (fully considering the future use, maybe we shall change the type of the filter, for which the current state is determined by many former steps’ states, like Kalman filter, in this case, we don’t want too many “contaminated” data), also in the final plot, the first 10 s’ data will not be showcased to avoid confusion of the audience.

The examples of the results of two modes are shown as above, Fig. W13-4 and Fig. W13-5.

1. Combine all the pervious communication work

Pending.

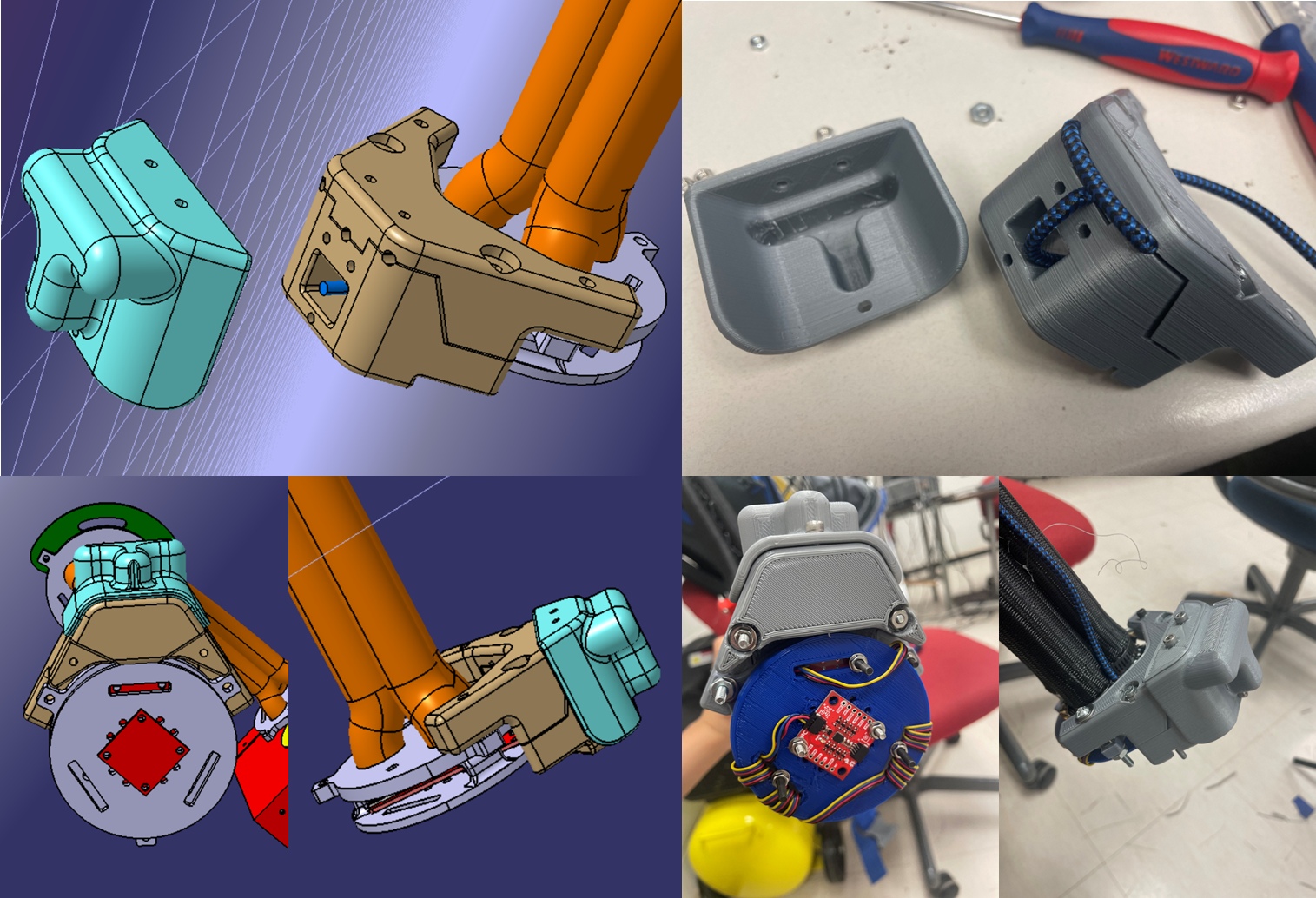
1. Auxiliary parts designed for backpack

To ensure the IMU and ESP32 can be mounted on the base of the arm properly, I designed another plate for it last week, it worked pretty well for the practical application, a comparison between the real object and 3D design is shown as follows.



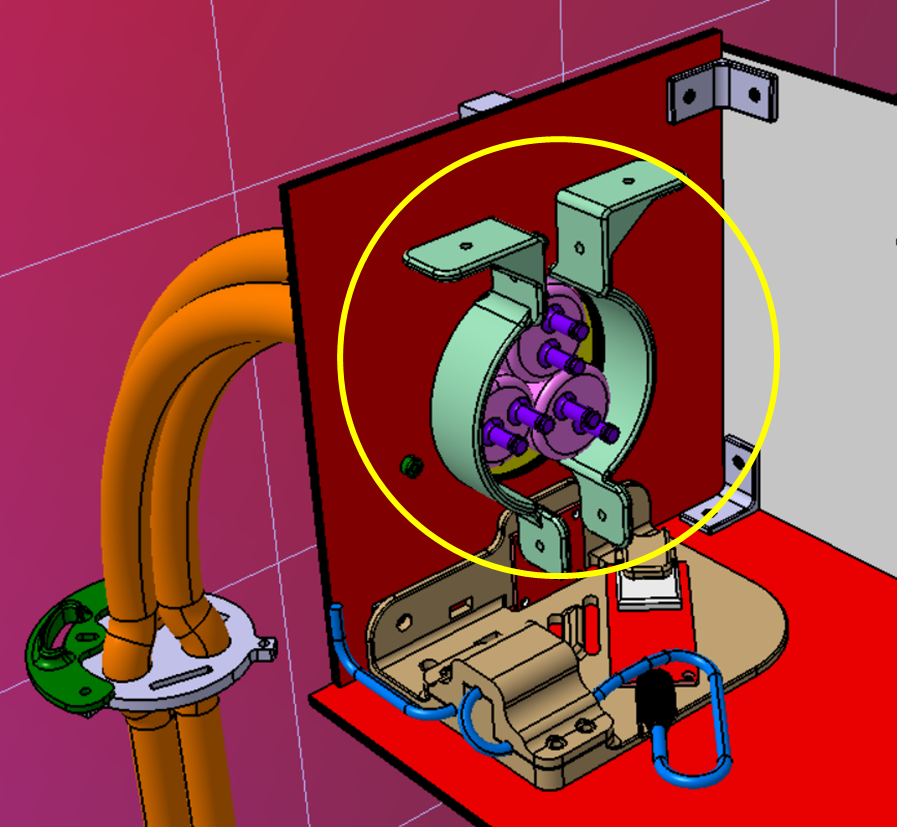
**Fig. W13-6** The comparison between the real object and 3D design of the IMU & ESP32 assembly plate mounted on the base of the arm

And also I have designed a “shield” for the port of ESP32 mounted on the end effector of the arm to protect it from collision during the experiment. The design also proved to be successful which can be seen below.



**Fig. W13-7** The comparison between real object and 3D design of the ESP32 port shield for the SRA end effector

Thirdly, to schedule the wiring route properly, we also need additional parts to fix the tunnel or channel for the wire to go through, which has been highlighted in the yellow circle in the figure below, now it’s still in alpha stage and it has not been printed out yet, because the dimension and positions of the big hole for the three pneumatic chambers to go through and the screws are not symmetric about the central line of the backpack, which increase the difficult of design. To increase the interchangeability of the parts, both sides need to be designed identically, I still need to verify the dimension.

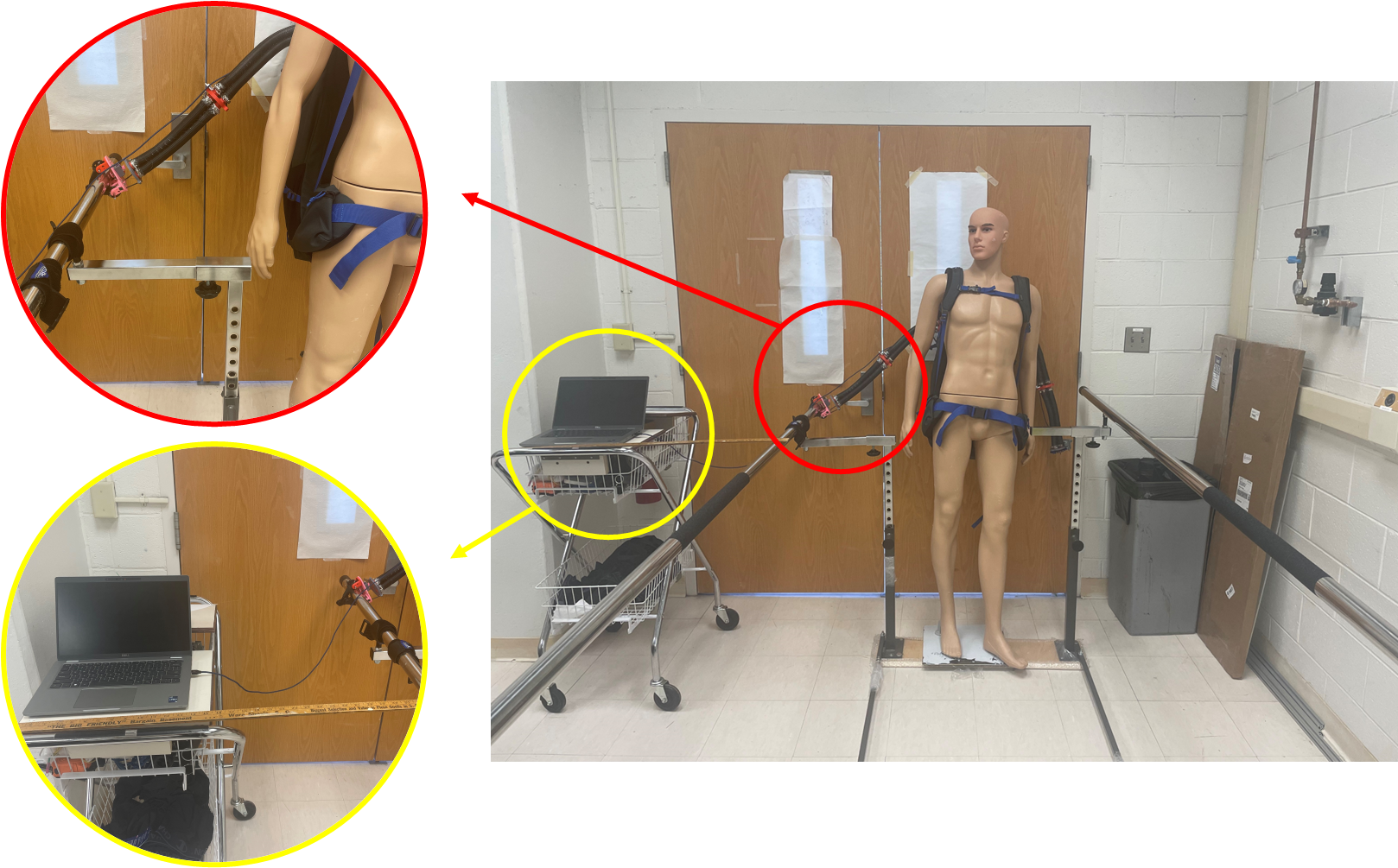


**Fig. W13-8** The conceptual design of the wire channel fixer

1. Experiment schedule for the mannequin falling down

The alpha version of the schedule will be focused on the safety issues and necessary actions need to be taken pre the experiment, I hope I can come up with a draft before the weekly meeting next week.

Besides, I have a rough idea about the safety distance away from the handrail and length of the wire we need according to the set up in the figure below. The mannequin was set to be in the centre of the handrail, to test the if the gripper could reach the handrail, when the handrail is fully stretched laterally, the gripper was just about to reach it, which means that in the real experiment, it is likely for the gripper to miss the handrail it we choose the arm to swing in the direction which is perpendicular to the handrail, we probably need to fix the arm swing strategy to let it swing with a certain angle.



**Fig. W13-9** Set up for the mannequin falling down experiment

Apart from this, to consider safety distance and extreme conditions like the SRAs failed to grasp the handrail causing the mannequin fall onto the ground, what will be the minimum wire length to ensure it will not destroy any electronic connections, it has been calculated that the wire needs to be at least 5 m long (6 will be better).