IVR Assignment

Maksymilian Mozolewski & Martin Lewis

November 2020

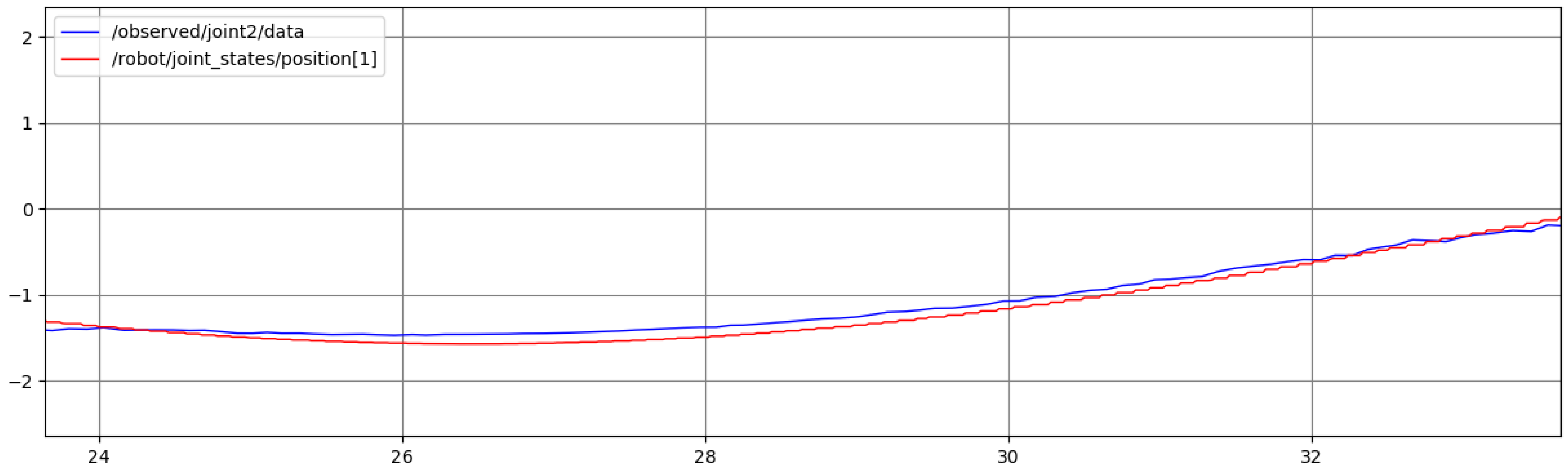

Figure 1: Joint2 Angle

2.1

The first part of this section was straight forward calculating the position of the angles based of the sinusoidal positions. This achieved with rospy's get time function and sin.

The next section is getting the 3D coordinates from the 2 2D images. This was done initially with a naive system that assumed that the camera was always orthogonal, which it isn't. This worked fine for most cases except the extreme cases where the robot arm would point directly into a camera. This was replaced by a more complex system considering the cameras to be simple pinhole cameras and reversing the projection into the 2D image back into the 3D world coordinates.

Finally having got the 3D world coordinates then the angles had to be found. We tried a few different methods, optimisation, projections but finally settling for using trigonometry. Using atan2 to find the values of joint 2 and 3. Joint 2 is found with atan on the y and z components of the vector between the blue and green blobs. Then the vector is rotated by the found angle and then atan is run on the x and z components for joint3. This does however mean that the uncertainly and error in the angle of joint 2 is passed to joint3.

You can see the two example sections of an rqt_plot graph from joints 2 and 3 in figures 1 and 2.

Joint 4 however is calculated by a projection. This means its not reliant on the first two angles. It works out the angle between the vector between the green and red blobs and the vector between the blue and green ones.

2.2

The target is distinguished first by a threshold over the colour orange. Then the cv2 match template function is applied to it using two templates (template-box.png and template-sphere-png both are in root folder of the repository) this returns a position in the image supplied to it. This is then turned into 3D world coordinates by the system discussed in 2.1. You can see a graph showing the results in figure 4. The major issue is that of the z value, it is frequently not quite correct, it usually in the correct ball park but not as accurate at the other two. X and Y however are very accurate for the vast majority of the time.

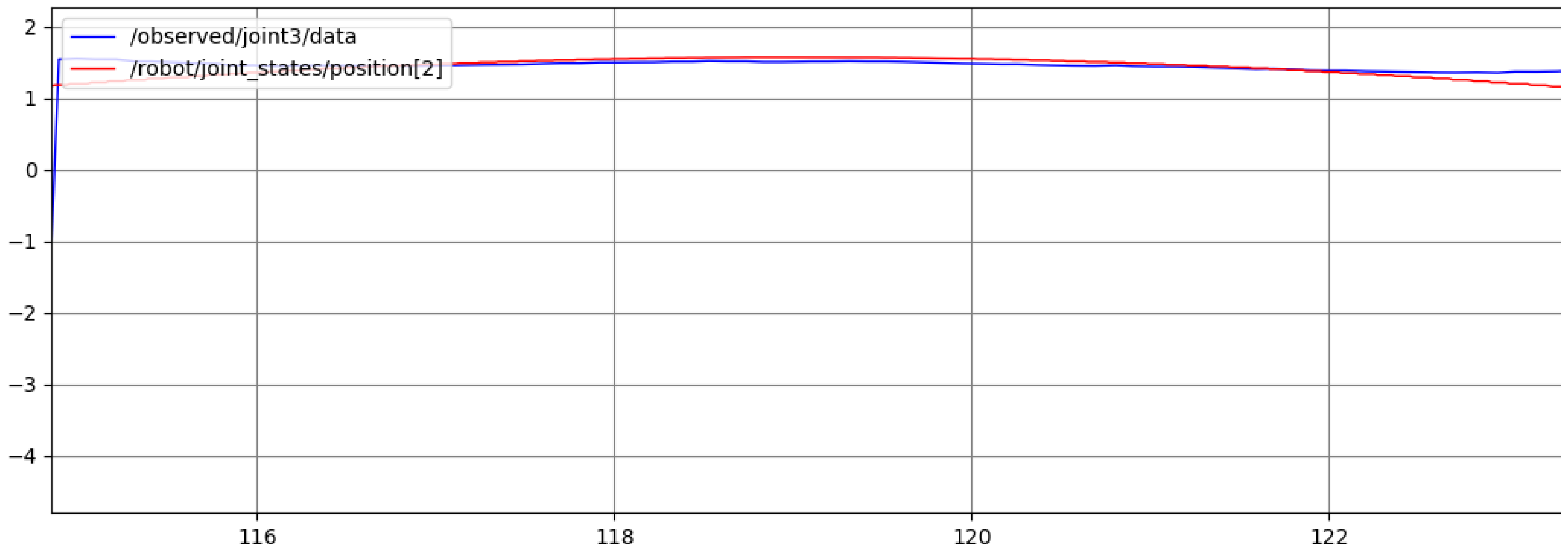

Figure 2: Joint3 Angle

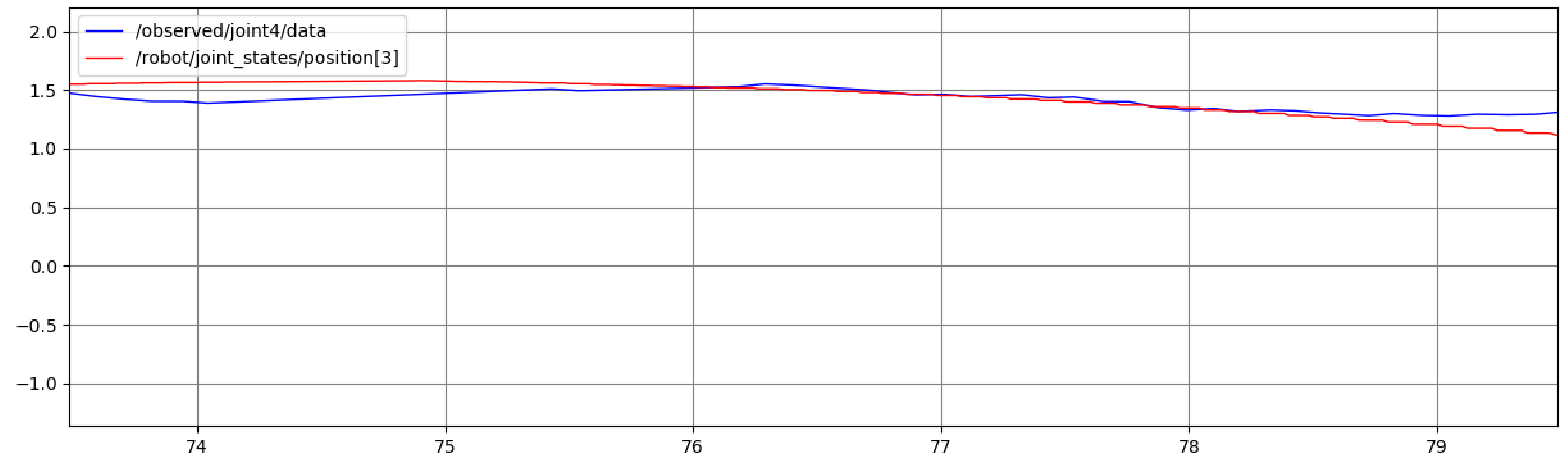

Figure 3: Joint4 Angle

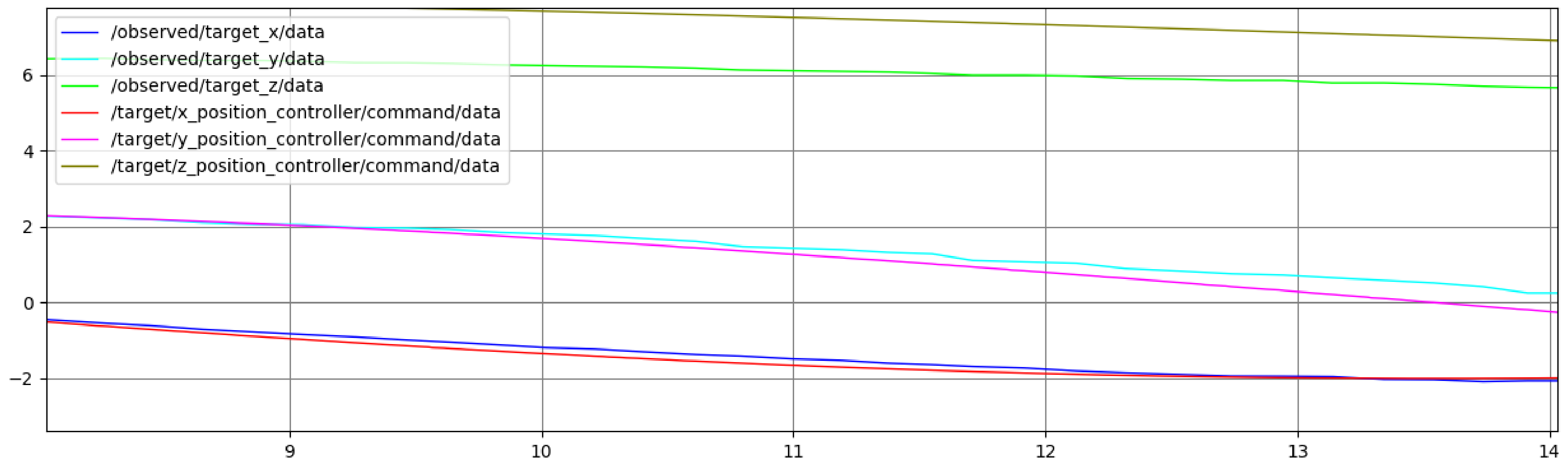

Figure 4: Target Position (/observed/...) and actual position (/target/...)

3.1

The forward kinematics, was derived using d-h parameters and a sympy script which parsed it to avoid any mathematical errors (The said script is available in the repository). The final forward kinematics translation component looks like the following:

$$\begin{bmatrix} x_e \\ y_e \\ z_e \end{bmatrix} = \left(\begin{matrix} 3\left(S_{q_1} S_{q_2} C_{q_3} + S_{q_3} C_{q_1}\right) C_{q_4} + 3.5 S_{q_1} S_{q_2} C_{q_3} + 3 S_{q_1} S_{q_4} C_{q_2} + 3.5 S_{q_3} C_{q_1} \\ 3\left(S_{q_1} S_{q_3} - S_{q_2} C_{q_1} C_{q_3}\right) C_{q_4} + 3.5 S_{q_1} S_{q_3} - 3.5 S_{q_2} C_{q_1} C_{q_3} - 3 S_{q_4} C_{q_1} C_{q_2} \\ -3 S_{q_2} S_{q_4} + 3 C_{q_2} C_{q_3} C_{q_4} + 3.5 C_{q_2} C_{q_3} + 2.5 \end{matrix}\right)$$

$$\begin{bmatrix} \dot{x_e} \\ \dot{y_e} \\ z_e \end{bmatrix} = \left(\begin{matrix} \dot{q_1}\left(\left(-3S_{(q_1)} S_{(q_3)} + 3S_{(q_2)} C_{(q_1)} C_{(q_3)}\right) C_{(q_4)} - 3.5 S_{(q_1)} S_{(q_3)} + 3.5 S_{(q_2)} C_{(q_1)} C_{(q_3)} + 3 S_{(q_4)} C_{(q_1)} C_{(q_2)}\right) + \dot{q_2}\left(-3S_{(q_1)} S_{(q_2)} S_{(q_4)} + 3S_{(q_1)} C_{(q_2)} C_{(q_3)} C_{(q_4)} + 3.5 S_{(q_1)} C_{(q_2)} C_{(q_3)}\right. \\ \dot{q_1}\left(\left(3S_{(q_1)} S_{(q_2)} C_{(q_3)} + 3S_{(q_3)} C_{(q_1)}\right) C_{(q_4)} + 3.5 S_{(q_1)} S_{(q_2)} C_{(q_3)} + 3S_{(q_1)} S_{(q_4)} C_{(q_2)} + 3.5 S_{(q_3)} C_{(q_1)}\right) + \dot{q_2}\left(3S_{(q_2)} S_{(q_4)} C_{(q_1)} - 3 C_{(q_1)} C_{(q_2)} C_{(q_3)} C_{(q_4)} - 3.5 C_{(q_1)} C_{(q_2)} C_{(q_3)}\right) \\ \dot{q_2}\left(-3S_{(q_2)} C_{(q_3)} C_{(q_4)} - 3.5 S_{(q_2)} C_{(q_3)} - 3 S_{(q_4)} C_{(q_2)}\right) + \dot{q_3}\left(-3S_{(q_3)} C_{(q_2)} C_{(q_4}\right. \end{matrix}\right.$$