1 Robot Vision

1.1 Joint State Estimation 1

1.1.1 Algorithm Overview

This section explains the algorithm for estimating the robot joint angles. Two separate, supporting processes are invoked to capture the images from camera 1 (orthogonal to the y axis) and camera 2 (orthogonal to the x axis). Using both processes the joint angles are determined using vector geometry. Knowing that $a \bullet b = |a||b| \cos\theta$ angle *theta* between two joints can be determined by calculating $\cos^{-1}(\frac{a \bullet b}{|a||b|})$.

The first step is determining the location of vectors a and b from the images. This was completed by using colour masking techniques to isolate the joints from the rest of the image. The center of each joint was estimated by calculating the X, Y moments of each binary image that was derived from the colour masking, which gave co-ordinates in pixel space. If this process failed due to camera obscufication then the co-ordinates captured by the previous successful image capture were used as a proxy for the joint location. Figure one provides some examples of this happening. Camera one

(a) Plot Comparing Changes to Joint 3 Against Baseline

(b) Plot Comparing Changes to Joint 4 Against Baseline

Figure 1: Examples of where joint detection would fail. When this happens the last previous joint recording was used in the algorithm.

tracks co-ordinates x and z, and camera two tracks co-ordinates y and z. Using this information the 3d location of the joint is estimated by taking the x and y co-ordinates from each camera, with the z co-ordinate being the average of the z co-ordinate returned by the two cameras. This gives a 3 dimensional vector spanning the $[X, Y, Z]$ space containing the location of each joint.

To identify the location of each joint it was subtracted from it's downstream equivalent (blue was subtracted from yellow and red subtracted from blue), and converted to meter basis via the following equation: $[X', Y', Z'] = [X, Y, Z] * (L/\sqrt{\sum_c} \in x, y, z(x_i - x_{i-1})^2$. L represents the lengh of the links between joints and coord i and coord i-1 is the upstream and downstream joint. Following the meter conversion the joint angles were determined according to slightly different formulaes as demonstrated below:

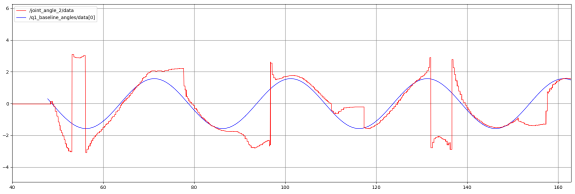
- **JOINT 2**: Tracks changes in the y axis between the yellow and blue joints. Because of this the scaled joint location is projected onto the identity x axis to find vector that is orthogonal to the x axis and joint 2. Joint 2 angle can be calculated as the angle between this projection and the y co-ordinate basis: $\cos^{-1}(\frac{PROJ \bullet [0,1,0]}{|PROJ| * |[0,1,0]|})$
- **JOINT 3**: Tracks changes in the x axis between the yellow and blue joints. Because of this the scaled joint location is projected onto the identity y axis to find vector that is orthogonal to the y axis and joint 2. Joint 3 angle can be calculated as the angle between this projection and the x co-ordinate basis: $\cos^{-1}(\frac{PROJ \bullet [1,0,0]}{|PROJ| * |[1,0,0]|}) - \pi$
- **JOINT 4**. Tracks the difference between the end effector and the blue joint in the y axis. As we're seeking the angle between these two vectors no projection onto the identity co-ordinate system is needed and the angle is $\cos -1(\frac{R \bullet Y}{|R| * |Y|})$

The inverse cosine implementation used is the numpy **arccos** method. As this does not take into consideration the signage of the vectors whether the rotation is positive or negative is calculated by reviewing the joint locations. For joints 2 and 4, in pixel basis, if the upstream x co-ordinate is to the

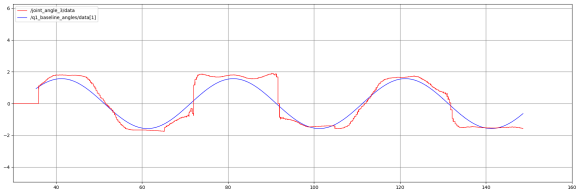
left of it's downstream joint then we know a negative rotation has occured and the angle is multiplied by -1 to reflect this fact. A similar process is taken by looking at the y co-ordinate for changes to the x axis for joint 3.

1.1.2 Results

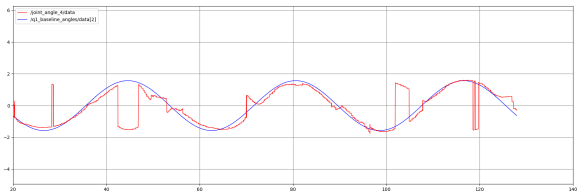
The algorithm was ran for 120 seconds with the recorded change in joint angle plotted against the estimated change in joint angle from the vision algorithm. Separate graphs are used to plot the deltas between the joint angles and the vision algorithm estimations. The results are shown in the plots below:



(a) Plot Comparing Changes to Joint 2 Against Baseline



(b) Plot Comparing Changes to Joint 3 Against Baseline



(c) Plot Comparing Changes to Joint 4 Against Baseline

Figure 2: The x axis denotes the algorithm run-time in float seconds. The y axis is used to plot the degrees in radians of rotation for each joint. The red curve is the estimated joint angle and the blue curve is the baseline joint angle.

1.1.3 Discussion

The joint angle detection algorithm is reasonable effective at tracking the changes to the joint states. For all the joints measured the velocities recorded by the algorithm roughly follow a sinusodial signal that's aligned with the changes to the joint angles.

The most accurate tracking occurs at joint 3. Whether this accuracy is a product of there only being one rotation on the x axis would require further empirical investigation but it's a useful starting explanation.

When the velocities of the joints 2 and 4 reach a maxima and minima the performance of the algorithm begins to deterorate. The most likely reason for this that when the joints these configurations the cameras are unable to gain a clear view of the joints and the previous measurements are used. This results in larger error between the baseline and the calculated joint angles. In figure 1a and 1c at approximately 60 seconds, 90 seconds and 100 seconds we see changes to the joint angle calculations that are perpendicular to the y axis. It is at these points that the algorithm cannot identify the joints due to there being a limited picture of the joints or no picture at all. Similar behaviour can be seen in joint 4 at approximately at 115 seconds into the algorithm and 40 seconds into the algorithm and by the fact that over the 120 second period the algorithm failed to isolate the joints 292 times, whereas 11 failures occured on the x axis.

As the number of processes that change joint angles on a particular axis increases the algorithm performance deterorates. This is because the combined processes put the joints into configurations that make colour detection difficult, which prevents the algorithm from identifying the joint locations and angles.