1 Robot Vision

1.1 Joint State Estimation 1

1.1.1 Algorithm Overview

This section briefly describes 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), camera 2 (orthogonal to the x axis). The images are published to image_topic1 and image_topic2. An additional process is setup to move robot joints 2, 3 and 4 in a sinusodial trajectory. The joints follow the following wavelengths. TD is the time delta in float seconds from the process beginning and the clock time which is determined by the **rospy.get_time()** method.

- Joint 1: $\frac{\pi}{2}\sin(\frac{pi}{15} * TD)$
- Joint 1: $\frac{\pi}{2}\sin(\frac{pi}{18} * TD)$
- Joint 1: $\frac{\pi}{2}\sin(\frac{pi}{18} * TD)$

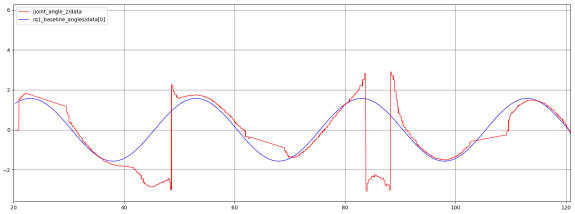
Once these processes are calculated the joint angle detection algorithm can be ran. The joint angle detection algorithm is based on the following algorithm:

procedure JOINT ANGLE DETECTION(camera_x, camera_y)
Ensure: image_x = camera_x
Ensure: image_y = camera_y
blob_yellow_x, blob_blue_x, blob_red_x = blob_detection(image_x)
blob_yellow_y, blob_blue_y, blob_red_y = blob_detection(image_y)
if blob_detection_successful **then**
distance_yellow_y = $3.2 * \sqrt{(blob\_yellow\_y - blob\_blue\_y)^2}$
distance_yellow_x = $3.2 * \sqrt{(blob\_yellow\_x - blob\_blue\_x)^2}$
distance_blue_x = $3.2 * \sqrt{(blob\_blue\_x - blob\_red\_x)^2}$
distance_blue_y = $3.2 * \sqrt{(blob\_blue\_y - blob\_red\_y)^2}$
distance_red_y = $2.8 * \sqrt{(blob\_red\_y - blob\_blue\_y)^2}$
determine pixel to meter; each blob = $blob * distance$
ja2 = $\tan(blob\_yellow\_y - blob\_blue\_y)$
ja3 = $\tan(blob\_yellow\_x - blob\_blue\_x)$
ja4 = $\tan(blob\_blue\_x - blob\_red\_x)$
publish joint2, joint3, joint4
else
Write image to disk for analysis
Print message joint angle could not be calculated
end if
end procedure

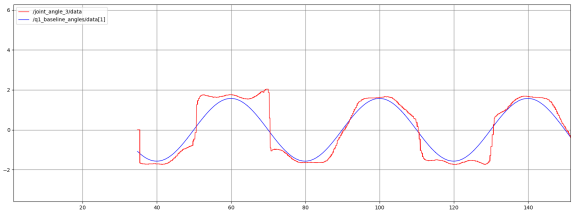
In working out the tangent the numpy atan2 library is utilized to ensure the correct angle quadrant is calculated. The joint locations are calculated using simple colour masking techniques to isolate the links, and then the center of each link is estimated via the moments of the image. For estimating changes to the joint angles the cameras facing the axis are used independently of one another, so changes to the joints in the y axis are tracked via the camera facing that plane and vice versa for camera x.

1.1.2 Results

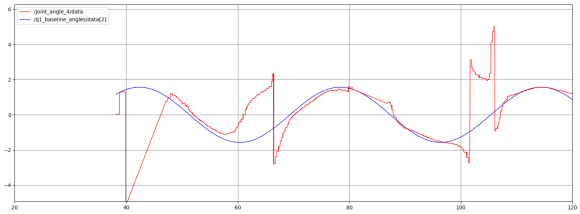
The algorithm was ran for 120 seconds with the recorded change in joint angle plotted against the estimated change in joint angle from the vision algorithm. Separate graphs are used to plot the deltas between the different joint angles and the vision algorithm estimations. The results are shown in the 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 1: 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 change in the joint angles and the blue curve is the change recorded by the algorithm.

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.

One explanation could be due to the delta between the joint movement signal being sent and the blob detection algorithm ingesting the images and calculating the angles. This though would create a more constant error delta with the algorithm similar to figure b.

A more likely explanation is that the velocities of joints 2 and 4 are not always aligned and performance deterorates most around the maxima or minima of the joint angle changes. When the velocity of one joint is approaching 0 the other can be either positive or negative. It is likely that this difference in joint change near the inflection points prevents the cameras getting a clear view of the joints, resulting in greater error recordings.