PROJECT REPORT

Robot Localization and Navigation

Submitted by:

Name: Harshavardhan Vibhandik
NET ID: hsv2015
NYU ID: N13023471

Spring 2024
Submitted on: April 7, 2024

ROB-GY6213 Robot Localization and Navigation

Instructor: Prof. Giuseppe Loianno

Introduction

In project 2, we have to estimate the position and the orientation of the quadrotor on the basis of April Tags. We have already provided the camera data in the given folders. We need to localize the robot by using the coordinates of April tags.

Explanation & Results

Part I:

In this part, we need to calculate the pose of the quadrotor. We already have provided the coordinates of the corners of April tags in the camera frame.
Firstly, we need to find the coordinates of the given April tags in the world frame.

We calculate the values of the centre and the four corners of the given April tags, namely, p0, p1, p2, p3, p4. The values of the id's are stored in the 'res' variable.

Then, we have to calculate the 'h' matrix using the below-mentioned formula:

$$\lambda_i \begin{pmatrix} x_i' \\ y_i' \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

And by calculating A by using the given equation:

$$Ah = 0$$

$$\begin{pmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i' \\ 0 & 0 & 0 & x_i & y_i & 1 & -y_i'x_i & -y_i'y_i & -y_i' \end{pmatrix} \underset{\uparrow}{\boxed{h}} = 0$$

The above matrix contains rows where xi and yi are the x and y-coordinates of every corner of the April Tags, and xi' and yi' are the image coordinates in the camera frame. Each matrix is formed for each corner and centre (po, p1, p2, p3, p4) for every id at that time stamp 't'. So, A is a matrix of all the IDs the camera sees. Using the 'svd'(Singular Value Decomposition) of A, we find the U, S, V matrix where 'h' is the 9th column of the V matrix. We have to reshape this and make this column in a 3x3 matrix.

We get its rotation and translation by multiplying the ‘h’ matrix by K^-1.

$$R_1^T R_2 = 0$$

|| R1|| = ||R2|| = 1

Then we find the svd(R),
R = [R1 R2 R1xR2];

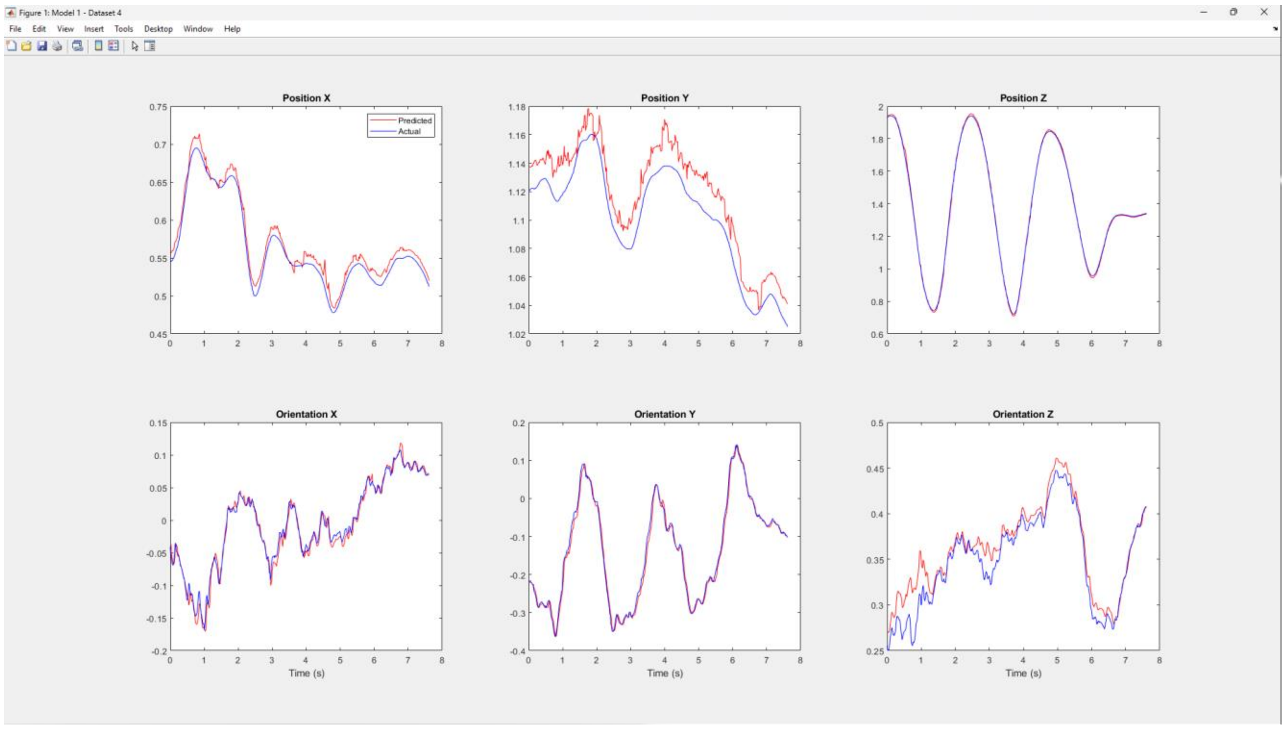
$$\begin{pmatrix}\hat{R}_1 & \hat{R}_2 & \hat{R}_1 \times \hat{R}_2\end{pmatrix} = USV^T$$

$$R = U \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(UV^T) \end{pmatrix} V^T$$

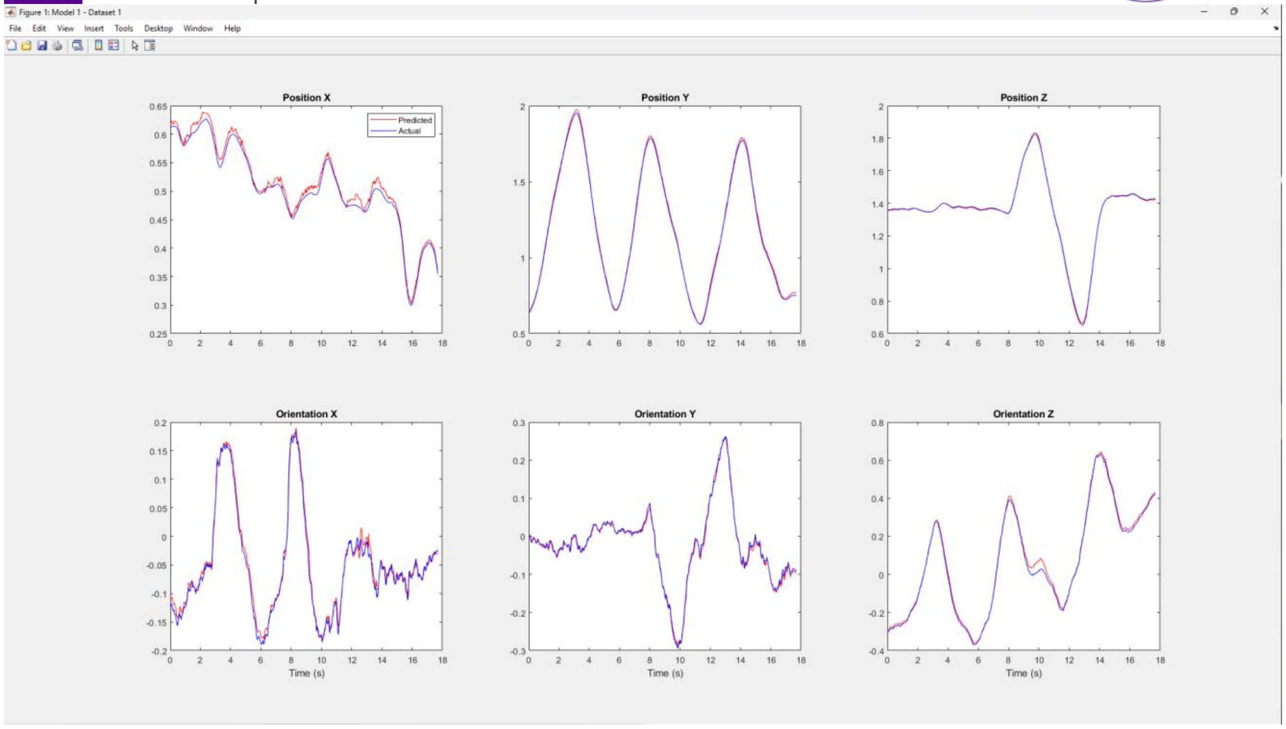
The translation vector of the camera with respect to the world in the camera frame is given by:

$$T = \hat{T}/\|\hat{R}_1\|$$

Results:
To get the results, we run the poseEstimation file, which then calls the plotData file:



Part1: Dataset: 4



Part 1: Dataset 1

Part II:

In this part, we need to calculate the optical flow of the corner points of the given images. Then, we estimate the robot's velocity with respect to the world frame.

For optical flow, a KLT tracker monitors these corner points within images.

The point locations in the camera frame are determined by multiplying the inverse of intrinsic K with a 'z' column in previous points.

We must calculate the optical flow by establishing a tracker to trace corner points from the previous image to the current. We then apply the estimatePose function to determine the output position, connecting the tracker's results to the velocity and the camera's position in the world frame.

$$\dot{\mathbf{p}} = \frac{1}{Z} A(\mathbf{p})\mathbf{V} + B(\mathbf{p})\mathbf{\Omega}$$

where A(p) , V , B(p) , omega are given correspondingly. These represent velocities and rotations, respectively.

For each corner, we generate matrices A and B in a loop and construct matrix H,

H = [1/Z*A(p),B(p)]

To find the Z value, representing the distance between corner points and the camera, we use the dot product between the translation vector of the image and corner point locations in the camera frame. Using the angle θ between two vectors, we find Z, representing the camera's position relative to the world frame.

Z = position of the camera in z-axis/cos (theta)

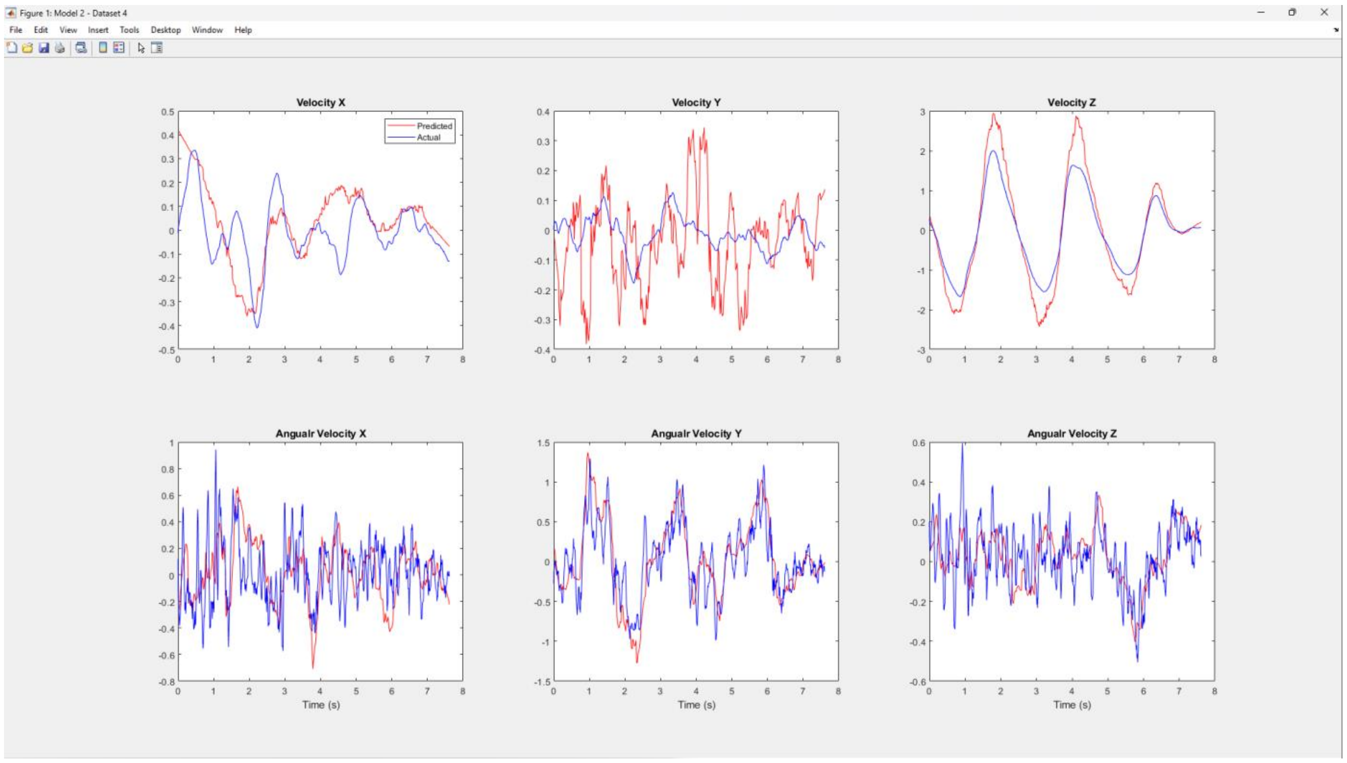
For velocity computation, we considered

dt = timestamp of current image−timestamp of the previous image

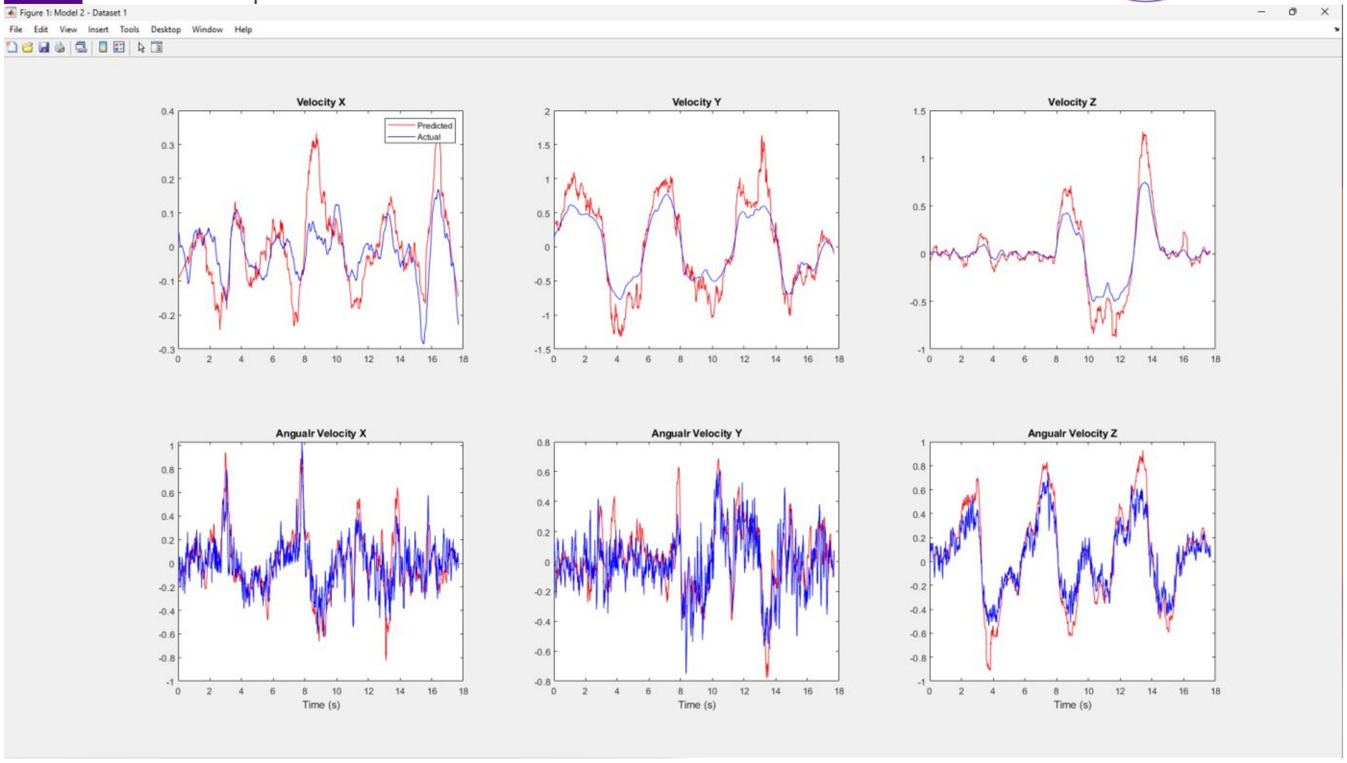
We then determine the camera velocity in the camera frame and translate it to world frame velocities using adjoint matrices and skew matrices. This transformation establishes the robot's velocity relative to the world.

We finally computed the robot's velocity relative to the world frame using optical flow. We visualize this by plotting the actual versus predicted velocities of the robot.

Results:
To get the results, we run the OpticalFlow.m file, which then calls the plotData file:



Part 2: Dataset: 4

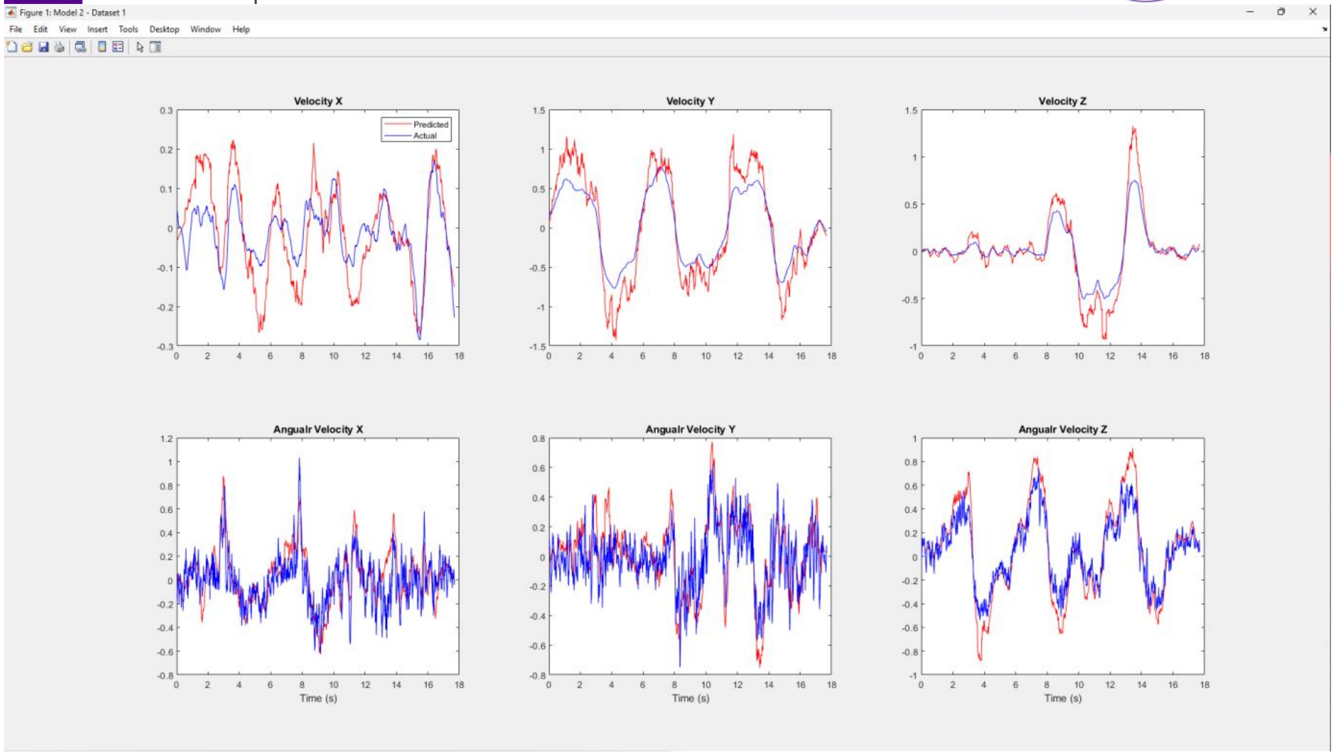


Part 2: Dataset: 1

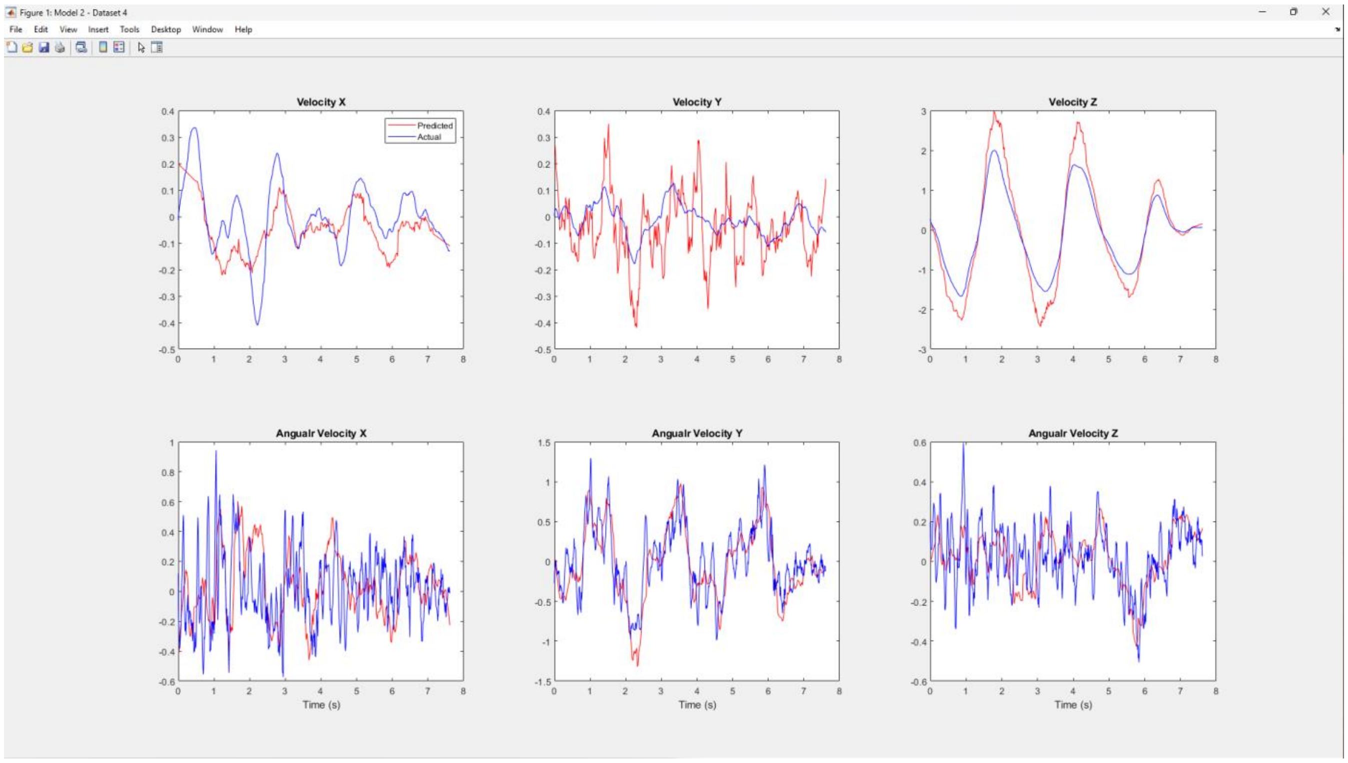
Using RANSAC:

Outliers can appear when we are calculating optical flow. To resolve this, we apply the RANSAC algorithm, which selectively samples three random values to compute the optical flow and identify and discard these outliers. This results in accurately determining the robot's velocity relative to the world frame.

Results:



Dataset: 1



Dataset: 4