Vision Based Pose Estimation

Name : Penmetsa Tejaswi Raju

NetID : pr2258

N Number : N18266865

Robot Localization and Navigation Project – 2

ROB-GY-6213

Professor Giuseppe Loianno

Penmetsa Tejaswi Raju
N18266865

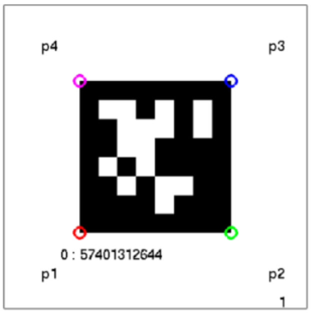
1. Introduction

In this project I have implemented a Vision based Pose estimation method, in which I am estimating the velocity in X, Y and Z and the angular velocity in X, Y and Z. There are two parts to this project in the first part which is on Pose estimation and the second part on Velocity Estimation with the implementation of RANSAC.

2. Pose Estimation (Part – 1)

2.1 Get Corner

In the getCorner function I am finding the coordinates of the corners and the center of all the April tags visible in the world frame when an id is passed to the function. The method that I am using to calculate is that I am considering the P4 point (according to the figure) for April Tag with ID 0 to have the coordinates (0,0) and the calculating the coordinates of all the points P4 for all April Tags that are being passed given in each ID. So, in every iteration all the coordinates i.e. P0 (Center), P1, P2, P3 and P4 is being calculated.



2.2 Estimate Pose

In this estimatePose function I am passing the structure named data which contains the timestamp (t), the roll pitch yaw (rpy), the angular velocity (omg), the acceleration (acc), the rectified image data (img), the ID of each April tag observed in the image (id) and finally all the points P0, P1, P2, P3 and P4 of each ID as expressed in image coordinates.

Firstly, I am collecting the corner data in world frame by passing the function getCorner and storing all the data in a variable named “ res ”.

Our basic objective from this function is to find the position and the orientation in the world frame with respect to the body frame. Thus to find this first we find the position and orientation in world frame with respect to camera frame by using the following equation,

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

Where we have to find the term $\begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}$[H matrix]. To find this we use the matrix,

$$\begin{pmatrix} x_i & y_i & 1 & -x_i'x_i & -x_i'y_i & -x_i' \\ 0 & 0 & 0 & -y_i'x_i & -y_i'x_i & -y_i' \end{pmatrix}$$

Then we are finding the matrix above for all the five points i.e. P0, P1, P2, P3 and P4, and then we are vertically concatenating this for all these points that we get from ID. After doing this I am calculating the SVD of the resulting matrix and we get the values of H matrix from the 9^{th} column of the term V we get by doing SVD and making it to a 3x3 matrix. After finding this I am changing the sign of this H matrix if the value of H is less than zero. We are then calculating the matrix $(\hat{R}_1 \quad \hat{R}_2 \quad \hat{T})$ by doing the following multiplication,

$$\begin{pmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}$$

Where $\begin{pmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{pmatrix}$ is the Camera matrix which has been provided to us.

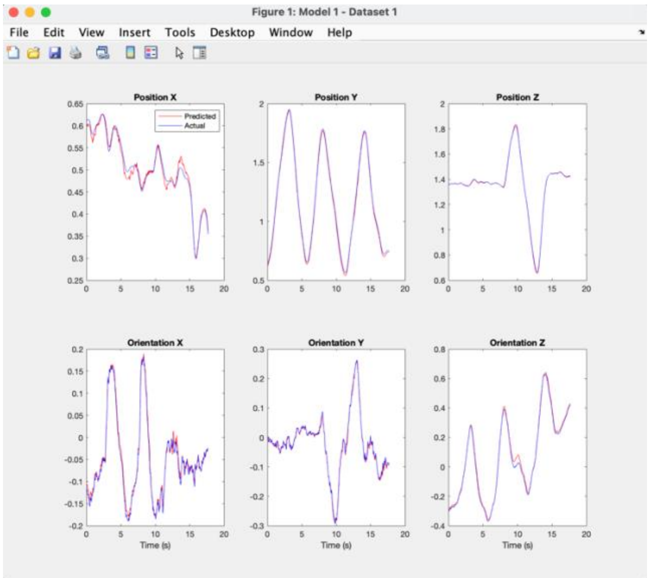
By doing this we find out the terms $(\hat{R}_1 \quad \hat{R}_2 \quad \hat{T})$, using these terms we are using doing the following step

$$\left(\hat{R}_1 \;\; \hat{R}_2 \;\; \hat{R}_1 \times \hat{R}_2\right) = USV^T$$

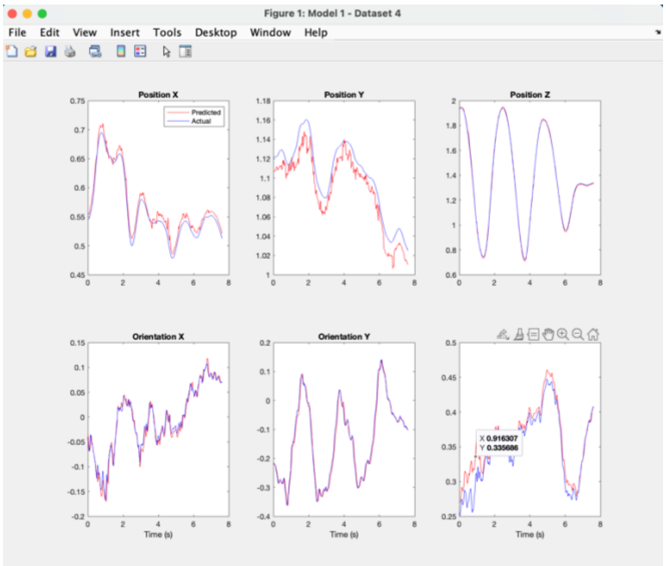
$$R = U \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(UV^T) \end{pmatrix} V^T$$

Here the R matrix that we get is the rotation matrix in camera frame with respect to the world frame. Then using the parameters that are given to us and the images provided to us we calculate the Transformation matrix of world frame with respect to the body frame. We then take out the position and orientation from this transformation matrix. For the orientation matrix we convert the rotational matrix to Euler angle in ZYX manner.

2.3 Results



Dataset 1



Dataset 4

3. Velocity Estimation (Part -2)

3.1 Optical Flow

In the file Optical Flow, I first take two images, one from the current time stamp and one from the previous time stamp. We are using these two images to compare the pixels so that we can find out the velocity and thus compare these values so that we can find out how the drone is being localized.

The point tracker that I am using is named MaxBidirectionalError. The object is that is specified to do the point tracker calculates the Bidirectional error. This is the value is the distance in pixels from the original position of the points to the final location after backward tracking. Then I am using the corner detector called as "detectFASTfeature" to detect the corners for all the points on the April tag and the points on the scribble that was on the April tag. Then I am storing the location of these points in the format of (x, y).
After this step I am initializing the tracker variable the location variable and the image that was passed i.e. the Last Image. Then afterwards I following the same steps to calculate locations of all the corners for the current image. Using these points I am applying the formula's to calculate the velocity of both x and y.

$$\dot{x} = \frac{dx}{dt} = \frac{Current\ Image\ Points - Old\ Image\ Points}{Time\ (n) - Time\ (n-1)}$$

$$\dot{y} = \frac{dy}{dt} = \frac{Current\ Image\ Points - Old\ Image\ Points}{Time\ (n) - Time\ (n-1)}$$

Then I am storing these values in a variable one on top of the other.

The next part is the point where we calculate the height, Here I am first running estimate pose function from part one where the output from the function is position, orientation and rotation matrix of camera to world frame. From the output I am calculating the Transformation matrix of camera to world frame then I am taking the (3,4) element out of the transformation matrix and considering this to be the height. Then the next step is that I am making the matrix $(A \quad B)$ where A and B is according to the professors slides. So, the final matrix looks like

$$\begin{pmatrix} -1/_{Z} & 0 & x & xy & -(1+x^2) & y \\ 0 & -1/_{Z} & y & 1+y^2 & -xy & -x \end{pmatrix}$$

3.2 Velocity calculation without RANSAC

To calculate velocity without RANSAC I am just taking the pseudo inverse of the matrix and multiplying it with the velocity that I have calculated. So I get a (6 x 1) matrix which has all the linear velocity and the angular velocity. Then I am converting the velocity from camera frame to world frame. And this velocity is passed so that it can be filtered and then plotted.

3.3 Velocity calculation with RANSAC

So, I am passing the Velocity, the position from camera frame, the height, rotation matrix from camera to world frame and the RANSAC hyper parameter. First, I am calculating the number of iterations that is needed by using the formula

$$k = \frac{\log(1 - p_{success})}{\log(1 - \in^{M})}$$

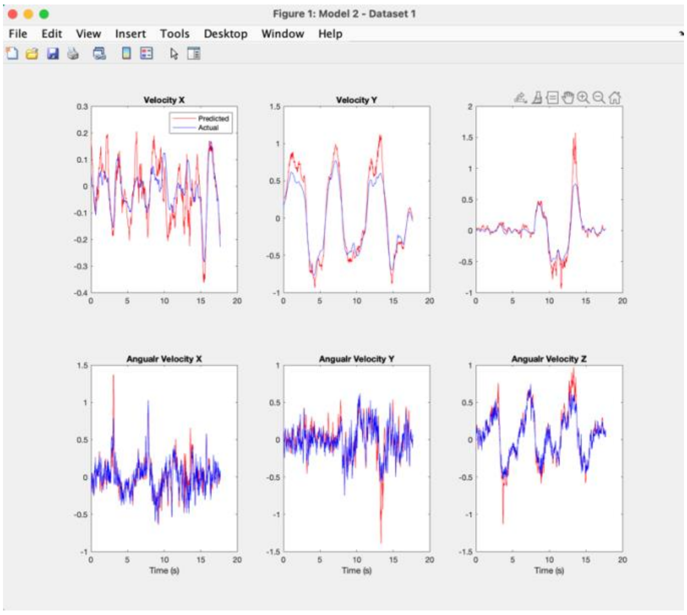
So, I am taking $p_{success}$ to be 0.99, $\in$ to be 0.8 and M to be 4 i.e. to be homographic so the value of k that I am getting is 9, thus that is 9 iterations to calculate the estimated velocity. I am taking the threshold where the inliners add up to be 1e-3. So, when it goes above the threshold the inliners increase by one. And at the end when the inliners are greater than the Maxinliners flag the Velocity calculated is being used as the output parameter.

After passing the velocity I am converting the velocity from camera to world frame. This velocity is then being filtered and then plotted.

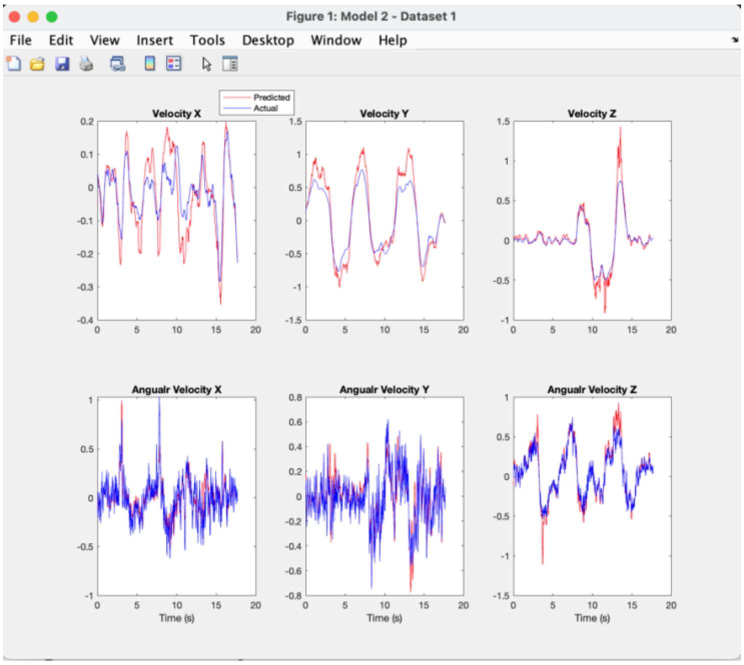
3.4 Filtering

I am filtering all the values that is being plotted both the time and the velocity matrix that is being for filtering the values I am using the MATLAB filtering algorithm named as “sgolayfilt”, which is called Savitzky – Golay Filtering and is used to generate a smoothened signal. The input parameters for this filtering is the function that needs to be filtered, the order of the equation and the number of terms that needs to be filtered together. The order of the equation that I have used is 1 and the number of terms taken input are 17 for velocity x, 17 for velocity y, 7 for velocity z, 5 for angular velocity x, 5 for angular velocity y and 5 for angular velocity z. Using these values to filter the graphs have smoothened and give a regular graph.

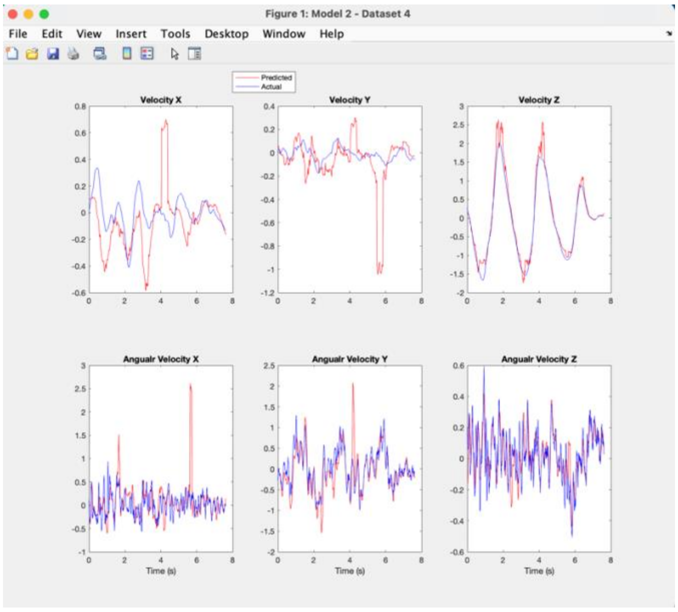
3.5 Results



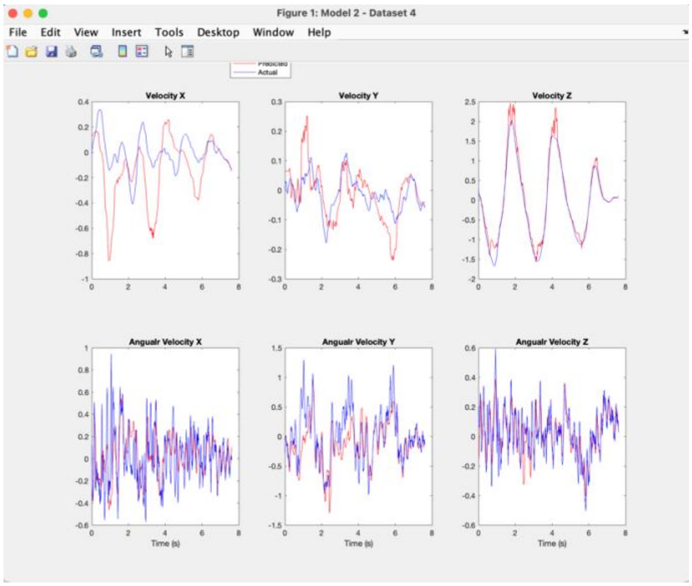
Dataset 1 with RANSAC



Dataset 1 without RANSAC



Dataset 4 with RANSAC



Dataset 4 without RANSAC

4. References

1. Lecture notes from the course Robot Localization and Navigation by Professor Giuseppe Loianno.
2. Introduction to Filtering from Vijay Kumar