# Project 2 Report

ROB-6213 Robot Localization and Navigation

Aditya Wagh (amw9425)

March 2022

# Contents

**Collaborators** : Pavan Posani, Mustafa B

# 1 Introduction

In this project, we implemented a vision based 3-D pose estimator which estimates position and orientation of the quadrotor based on AprilTags. [1] We also estimated the linear and angular velocity using the optical flow between subsequent images. We used the KLT [2] tracker to compute the optical flow between two images.

# 2 Camera Calibration

The camera projection matrix, P consists of two parts: intrinsic and extrinsic. The extrinsic part $[R \mid t]$ consists of the pose of the camera and the intrinsic part consists of camera calibration matrix K.

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = P \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \tag{1}$$

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = K[R \mid t] \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \tag{2}$$

P encodes a 3D to 2D transformation between world points and points in the image. The camera calibration matrix K is defined in equation (3)

$$K = \begin{pmatrix} f & 0 & x_0 \\ 0 & f & x_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3}$$

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \tag{4}$$

# 3 Pose Estimation

We used the 4 point algorithm to find the pose of the Nano+ quadcopter. The world point correspondences were calculated using the mat coordinate system given in the handout. We got the coordinates for 5 points of the AprilTag, nnamely, the center $p_0$, bottom left $p_1$, bottom right $p_2$, top right $p_3$ and top left $p_4$. Using the sampled data given to us in the data folder, we got the position of these points in the images taken using the quadcopter.

Following this, we estimated the planar homography H, which maps the planar world points with points in the image.

$$\lambda_i \begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \tag{5}$$

Since the image point is dual rank enforces only two constraints, some algebraic tricks later, we get

$$\begin{pmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x_i' x_i & -x_i' y_i & -x_i' \\ 0 & 0 & 0 & x_i & y_i & 1 & -y_i' x_i & -y_i' y_i & -y_i' \end{pmatrix} h = 0 \tag{6}$$

For each of the AprilTag we find in the image, we stack the equations for each point in the AprilTags detected in the whole image. We get a matrix $A$ which has the dimensions 10n x 2, where 5 is the number of AprilTags found in the image. We then solve the equation $Ah = 0$ by computing the SVD of A. The matrix a can be decomposed into a three matrices U , S and V.

$$A = USV^T \tag{7}$$

We want to find the eigenvector corresponding to the smallest eigenvalue of V, which is the last column of V. Therefore, $h = V_9$. We then reshape h to get the homography matrix H.

$$H = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \tag{8}$$

Once we get the homography matrix, we can decompose it to get the rotation and translation as follows.

$$\begin{pmatrix} \hat{R}_1 & \hat{R}_2 & \hat{T} \end{pmatrix} = \begin{pmatrix} \hat{r}_{11} & \hat{r}_{12} & \hat{t}_1 \\ \hat{r}_{21} & \hat{r}_{22} & \hat{t}_2 \\ \hat{r}_{31} & \hat{r}_{32} & \hat{t}_3 \end{pmatrix} = \begin{pmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \tag{9}$$

However, we have the constraint that $\hat{R}_1^T \hat{R}_2 = 0$ and $||\hat{R}_1|| = ||\hat{R}_2|| = 1$ So we need to solve the following optimization

$$\arg \min_{R \in \mathrm{SO}(3)} \left\| R - \begin{pmatrix} \hat{R}_1 & \hat{R}_2 & \hat{R}_1 \times \hat{R}_2 \end{pmatrix} \right\|^2 \tag{10}$$

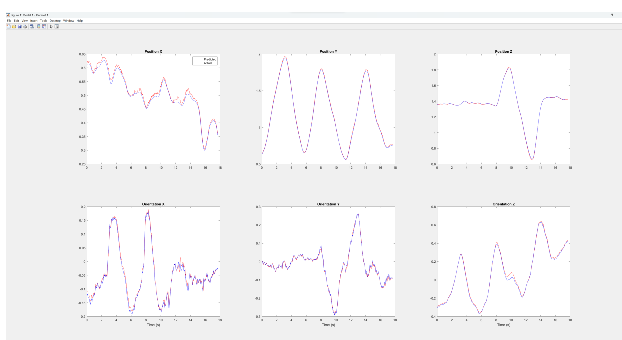

Figure 1: Pose Estimation Results for Dataset 1

The solution for the following optimization is

$$\begin{pmatrix} \hat{R}_1 & \hat{R}_2 & \hat{R}_1 \times \hat{R}_2 \end{pmatrix} = USV^T \tag{11}$$

$$R = U \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det\left(UV^T\right) \end{pmatrix} V^T \tag{12}$$

For dataset 1, the estimated pose and orientation are very close to our ground truth data.

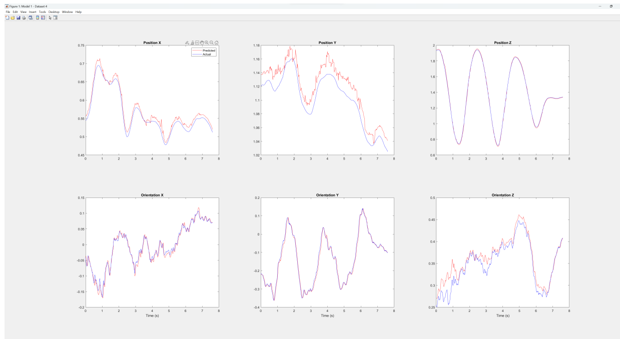

Figure 2: Pose Estimation Results for Dataset 4

For dataset 4, the estimated pose and orientation are also very close to our ground truth data, except for X & Y coordinate of position and yaw(Z) angle of the orientation which have a significant deviation from ground truth data.

# 4 Velocity Estimation

We estimated the velocity using the optical flow between successive frames captured by the drone at regular intervals. We assume that there are no illumination changes between successive frames, and there are minimal geometric deformations. things stay in sight To get good keypoints from the image to compute optical flow, we used the built in extractors in the Computer Vision Toolbox in MATLAB. We used the KLT tracker to track the points between successive frames. Using the time difference between the measurements and the positions of points in subsequent frames, we computed the optical flow. Using the optical flow, we estimated the velocity using the equation mentioned below.

$$\begin{pmatrix} u \\ v \end{pmatrix} = \frac{1}{Z}\begin{pmatrix} -1 & 0 & x \\ 0 & -1 & y \end{pmatrix}\begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix} + \begin{pmatrix} xy & -\left(1+x^2\right) & y \\ 1+y^2 & -xy & -x \end{pmatrix}\begin{pmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{pmatrix} \tag{13}$$

Here, $V_x, V_y, V_z$ is the linear velocity and $\Omega_x, \Omega_y, \Omega_z$ is the angular velocity of the body. $u$ and $v$ are the optical flows in $x$ and $y$ direction, and $x$ and $y$ are point coordinates in the image frame.

When the depth Z is known, we can solve this equation using least squares. The velocity calculated in this way might have outliers, so we need to use an outlier rejection method like RANSAC.

$$\dot{\mathbf{p}} = \frac{1}{Z}A(\mathbf{p})\mathbf{V} + B(\mathbf{p})\mathbf{\Omega} \tag{14}$$

$$\mathbf{H} = \begin{pmatrix} \frac{1}{Z_1}\boldsymbol{A}\left(\boldsymbol{p}_1\right) & \boldsymbol{B}\left(\boldsymbol{p}_1\right) \\ \vdots & \\ \frac{1}{Z_n}\boldsymbol{A}\left(\boldsymbol{p}_n\right) & \boldsymbol{B}\left(\boldsymbol{p}_n\right) \end{pmatrix} \tag{15}$$

$$\begin{pmatrix} \mathbf{V}^* \\ \mathbf{\Omega}^* \end{pmatrix} = \mathbf{H}^{\dagger}\dot{\mathbf{p}} \tag{16}$$

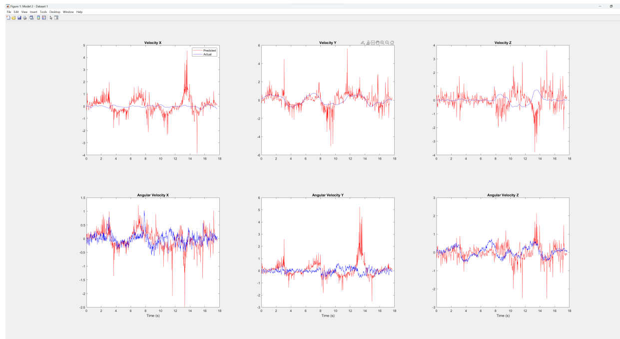

Figure 3: Velocity Estimation Results for Dataset 1 (No RANSAC)


Figure 4: Velocity Estimation Results for Dataset 4 (No RANSAC)

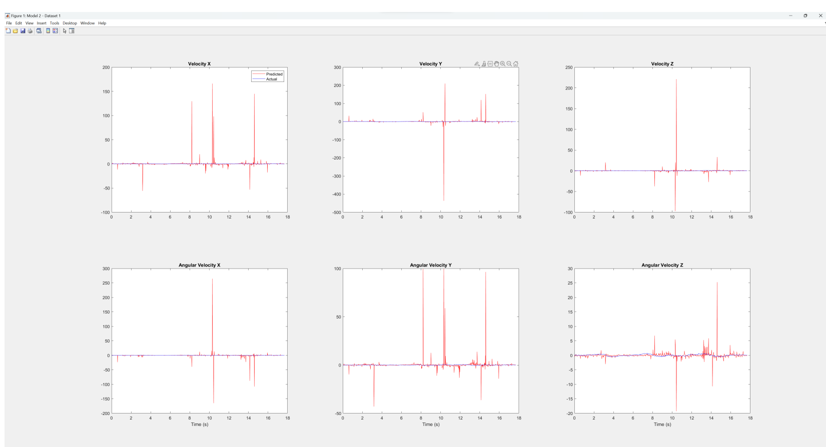


Figure 5: Velocity Estimation Results for Dataset 1 (With RANSAC)

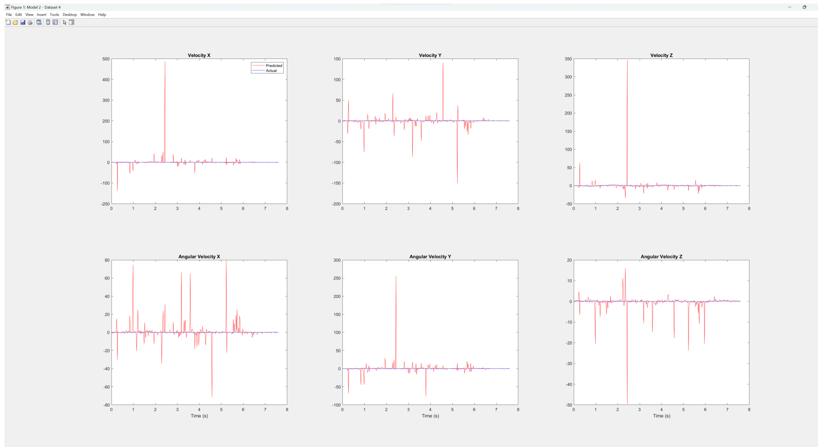


Figure 6: Velocity Estimation Results for Dataset 4 (With RANSAC)

# References

[1] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407. IEEE, May 2011.

[2] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, page 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.