Perception for Autonomous Robots-ENPM673

Report

By:

1. Sanket Acharya

2. Mithun Bharadwaj

3. Abhiram Dapke

INDEX

1. VISUAL ODOMETRY

Visual odometry is the process of determining the position and orientation of a robot by analyzing the associated camera images. odometry is the use of data from the movement of actuators to estimate change in position over time through devices such as rotary encoders to measure wheel rotations. While useful for many wheeled or tracked vehicles, traditional odometry techniques cannot be applied to mobile robots with non-standard locomotion methods, such as legged robots.

Most existing approaches to visual odometry are based on the following stages.

1. Acquire input images: using either single cameras. stereo cameras, or omnidirectional cameras.

2. Image correction: apply image processing techniques for lens distortion removal, etc.

3. Feature detection: define interest operators, and match features across frames and construct optical flow field.

a. Use correlation to establish correspondence of two images, and no long term feature tracking.

b. Feature extraction and correlation.

c. Construct optical flow field (Lucas–Kanade method).

4. Check flow field vectors for potential tracking errors and remove outliers.

5. Estimation of the camera motion from the optical flow.

Choice 1: Kalman filter for state estimate distribution maintenance.

Choice 2: find the geometric and 3D properties of the features that minimize a cost function based on the re-projection error between two adjacent images. This can be done by mathematical minimization or random sampling.

6. Periodic repopulation of trackpoints to maintain coverage across the image.

2. SLAM- Simultaneous localization and mapping

In navigation, robotic mapping and odometry for virtual reality or augmented reality, simultaneous localization and mapping (SLAM) is the computational problem of constructing or updating a map of an unknown environment while simultaneously keeping track of an agent's location within it. While this initially appears to be a chicken-and-egg problem there are several algorithms known for solving it, at least approximately, in tractable time for certain environments. Popular approximate solution methods include the particle filter, extended Kalman filter, Covariance intersection, and GraphSLAM.

3. RANSAC- Random Consensus Sampling

Random sample consensus (RANSAC) is an iterative method to estimate parameters of a mathematical model from a set of observed data that contains outliers, when outliers are to be accorded no influence on the values of the estimates. Therefore, it also can be interpreted as an outlier detection method.[1] It is a non-deterministic algorithm in the sense that it produces a reasonable result only with a certain probability, with this probability increasing as more iterations are allowed.

4. EIGHT POINT ALGORITHM

The eight-point algorithm is an algorithm used in computer vision to estimate the essential matrix or the fundamental matrix related to a stereo camera pair from a set of corresponding image points.

The basic eight-point algorithm is here described for the case of estimating the essential matrix. It consists of three steps. First, it formulates a homogeneous linear equation, where the solution is directly related and then solves the equation, taking into account that it may not have an exact solution. Finally, the internal constraints of the resulting matrix are managed.

The constraint defined by the essential matrix E is

$$(\mathbf{y}')^T \mathbf{E}\, \mathbf{y} = 0$$

for corresponding image points represented in normalized image coordinates y,y'. The problem which the algorithm solves is to determine E for a set of matching image points. In practice, the image coordinates of the image points are affected by noise and the solution may also be over-determined which means that it may not be possible to find E which satisfies the above constraint exactly for all points. This issue is addressed in the second step of the algorithm.

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ 1 \end{pmatrix} \quad \text{and} \quad \mathbf{y}' = \begin{pmatrix} y'_1 \\ y'_2 \\ 1 \end{pmatrix} \quad \text{and} \quad \mathbf{E} = \begin{pmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{pmatrix}$$

The constraint can be re-written as:

$$y'_1 y_1 e_{11} + y'_1 y_2 e_{12} + y'_1 e_{13} + y'_2 y_1 e_{21} + y'_2 y_2 e_{22} + y'_2 e_{23} + y_1 e_{31} + y_2 e_{32} + e_{33} = 0$$

where:

$$\tilde{\mathbf{y}} = \begin{pmatrix} y'_1 y_1 \\ y'_1 y_2 \\ y'_1 \\ y'_2 y_1 \\ y'_2 y_2 \\ y'_2 \\ y_1 \\ y_2 \\ 1 \end{pmatrix} \quad \text{and} \quad \mathbf{e} = \begin{pmatrix} e_{11} \\ e_{12} \\ e_{13} \\ e_{21} \\ e_{22} \\ e_{23} \\ e_{31} \\ e_{32} \\ e_{33} \end{pmatrix}$$

that is, e represents the essential matrix in the form of a 9-dimensional vector and this vector must be orthogonal to the vector y which can be seen as a vector representation of the 3*3 matrix y, y(transpose).

Each pair of corresponding image points produces a vector y. Given a set of 3D points P this corresponds to a set of vectors and all of them must satisfy

$$\mathbf{e} \cdot \tilde{\mathbf{y}}_k = 0$$

This means that e is the solution to a homogeneous linear equation.

5. PROCEDURE

Data Preparation:

The input images are in Bayer format. We used an inbuilt function from Opencv library to recover the color images.

img=cv2.cvtColor(img, cv2.COLOR_BAYER_GR2RGB)

We extracted the camera parameters using ReadCameraModel.py and undistorted the current frame and next frame using UndistortImage.py.

Pipeline:

1.We found **point correspondences** between successive frames by using **orb feature detection**.
2.We **used brute-force approach for matching** the features.

3. Estimated fundamental matrix using **the Eight Point Algorithm with RANSAC**.

4. We assumed that the camera obeys the pinhole model.

5. We **estimated essential matrix** from the Fundamental matrix by using the following equation.

$E = K^T F K$ where K is the camera matrix and F is the fundamental matrix.

6. We obtained Translation and Rotation matrices from Essential matrix for four camera poses.

7. After that, we performed **linear triangulation points** to get the 3d points. We used **the least square solution through SVD** to get the triangulated points given 2 camera poses. After this step we have 4 sets of triangulated points for each estimated pose.

$$\begin{bmatrix} y\boldsymbol{p}_3^\top - \boldsymbol{p}_2^\top \\ \boldsymbol{p}_1^\top - x\boldsymbol{p}_3^\top \\ y'\boldsymbol{p}_3'^\top - \boldsymbol{p}_2'^\top \\ \boldsymbol{p}_1'^\top - x'\boldsymbol{p}_3'^\top \end{bmatrix} \boldsymbol{X} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{A}\boldsymbol{X} = \mathbf{0}$$

[U, S, V] = svd(A)
X = V(:, end)

8. We narrow down to one of the configurations by disambiguating it using the **cheirality condition** i.e. unconstructed points must be in front of the cameras. A 3D point X is in front of the camera iff: **r3(X-C)>0,** where r3 is the third row of the rotation matrix.

8. This gives us the relative Rotation and Translation matrices between the two corresponding frames. We then calculate the global rotation and translation which gives the trajectory to be plotted.

6. SCREENSHOTS OF OUTPUIS:

FIGURE: Results using inbuilt OpenCV functions

FIGURE: Results using inbuilt OpenCV functions

FIGURE: Results using inbuilt OpenCV functions

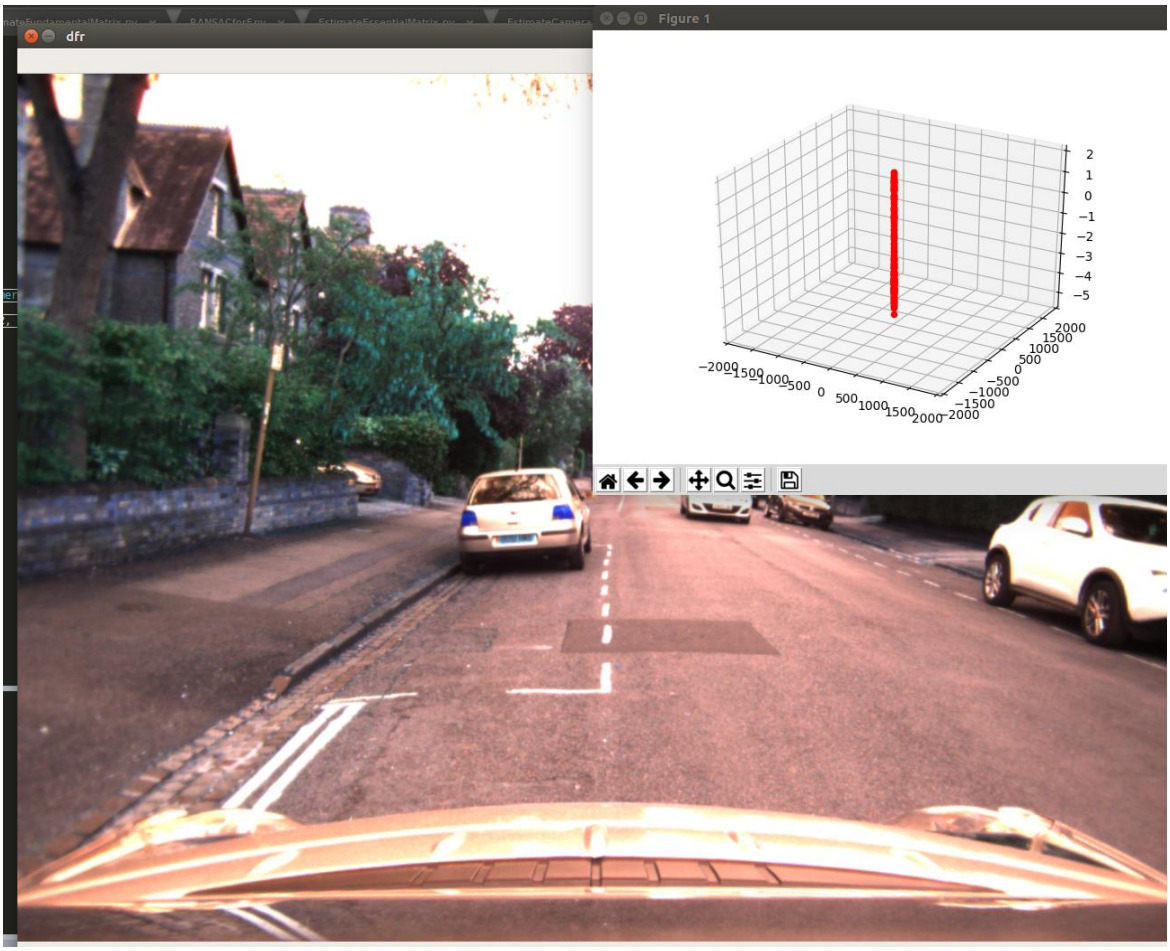

FIGURE: Results using Custom functions

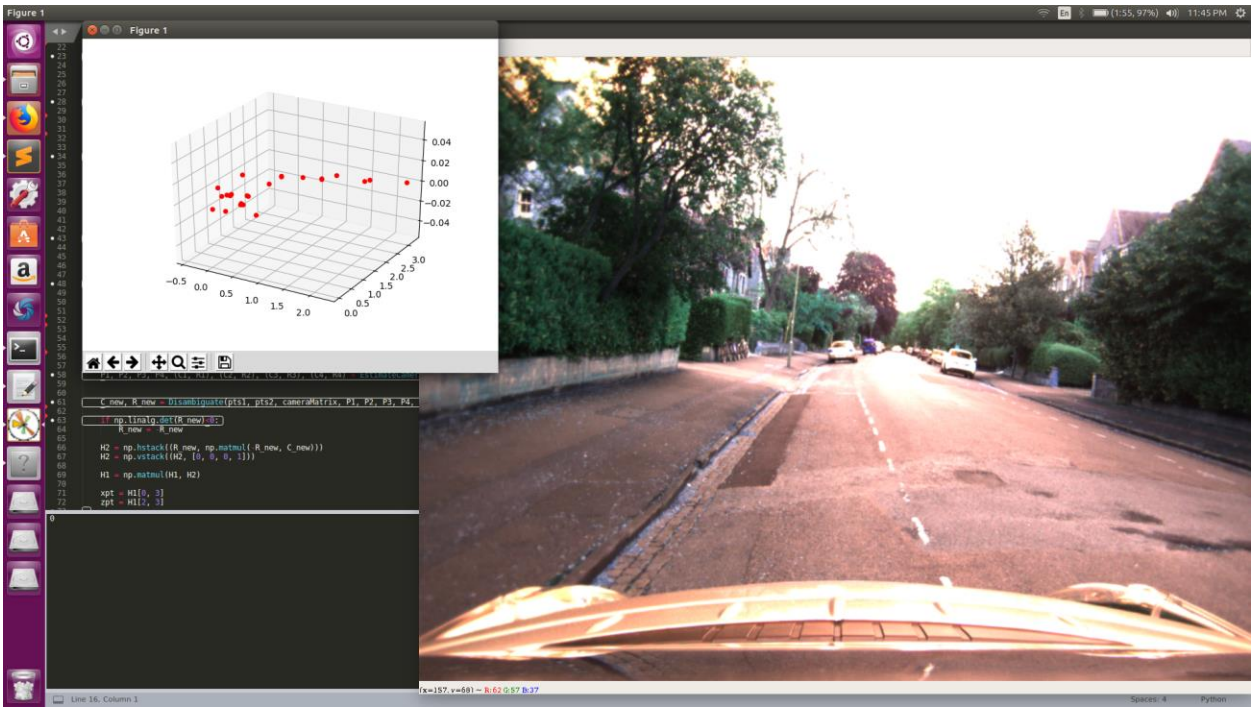

FIGURE: Results using Custom functions

7. REFERENCES:

1. Buildings built in minutes – An SfM Approach

2. Computer Vision ppt on The Eight Point algorithm– Penn State University

3. SLAM from Probabilistic Robotics