"[ENPM 673] Project-5"

Perception for Autonomous Robots

PROJECT REPORT

Nevil Patel -116897068

Shesh Mali-116831055

Akshay kurhade-116914529

VISUAL ODOMETRY

Table of Contents

Abstract 3
Implementation Flow 4
Data Preparation: 4
Finding Features: 4
Estimation of Fundamental and Essential Matrix: 5
Check For cheirality constraint: 8
Output Comparison (Extra Credits): 9
Non-linear Triangulation (Extra Credits): 10
Perspective-n-Point (Extra Credits): 10
Challenges Faced: 10
References: 11

Abstract

In this project of Visual odometry we are given a set of frames of driving sequence which is taken by the camera in car. By knowing the camera’s intrinsic parameters, we have estimated 3d motion of camera using structure from motion algorithm which tells about the relation of same point in two consecutive frames. Using the data, we have plotted the trajectory of the camera. This concept is very important in Robotics for navigation using SLAM. Using it we can map the feature points and descriptors around the scene to make a virtual environment of the scene.

Implementation Flow

Data Preparation:

A. Mosaic to RGB: Converted the Bayer pattern encoded image to a colour image using opencv functions

B. Undistort Image: Using undistortimage.py we undistorted the image which gives us a better image quality that is used for feature and descriptors detector used in our structure from motion algorithm.

C. Convert Image to Grayscale: We convert the undistorted image to gray scaled image because feature detection requires a gray scaled image.

D. Equalization of Image: By using Histogram equalization we made optimal brightness and contrast to image which is then a perfect image for feature detection.

Finding Features:

ORB is a good alternative to SIFT and SURF in computation cost, matching performance and mainly the patents. SIFT and SURF are patented and you are supposed to pay them for its use. ORB is basically a fusion of FAST key point detector and BRIEF descriptor with many modifications to enhance the performance. First it use FAST to find key points, then apply Harris corner measure to find top N points among them. It also use pyramid to produce multiscale-features. It computes the intensity weighted centroid of the patch with located corner at centre. The direction of the vector from this corner point to centroid gives the orientation. To improve the rotation invariance, moments are computed with x and y which should be in a circular region of radius r, where r is the size of the patch.

Estimation of Fundamental and Essential Matrix:

Calculation of F matrix: is done by first normalizing by shifting all the points around the mean of the points and enclosing them at a distance of sqrt (2) from the new origin or the new center. The F matrix is only an algebraic representation of epipolar geometry and can both geometrically (contructing the epipolar line) and arithematically. As a result, we obtain: x' TiFxi=0 where i=1,2,....,m. This is known as epipolar constraint or correspondance condition (or Longuet-Higgins equation). Since, F is a 3×3 matrix, we can set up a homogenrous linear system with 9 unknowns:

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_mx'_m & x_my'_m & x_m & y_mx'_m & y_my'_m & y_m & x'_m & y'_m & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{21} \\ f_{31} \\ f_{12} \\ f_{22} \\ f_{32} \\ f_{13} \\ f_{23} \\ f_{33} \end{bmatrix} = 0$$

Steps used to calculate F are:

1. Compute F from Normalized points

2. Force F to have rank 2

3. Undo normalization

8 Point Algorithm: In order to get the best inliers,if the distance of the key point from the epipolar line is lesser than one pixel we consider it as a good match. Hence it is considered as an inlier. The Sampson's formula for calculating the error is given below:

Epipolar line

$$e = \frac{|ax+by+c|}{\sqrt{a^2+b^2}} = \frac{|\mathbf{x}_2^T\mathbf{l}|}{\sqrt{\mathbf{l}_1^2+\mathbf{l}_2^2}} = \frac{|\mathbf{x}_2^T\mathbf{F}\mathbf{x}_1|}{\sqrt{(\mathbf{F}_1\mathbf{x}_1)^2+(\mathbf{F}_2\mathbf{x}_1)^2}}$$

The entire process of selecting the 8 random points and finding inliers is repeated multiple times. We have repeated the process for 50 iterations. We get the new fundamental matrix just by using the best inliers points.

Here We have used the Zhang's method of obtaining the 8x8 grid from the image. Then by randomly choosing the grids, and by randomly choosing a point inside this grid (if there are multiple points) we have selected the 8 points.

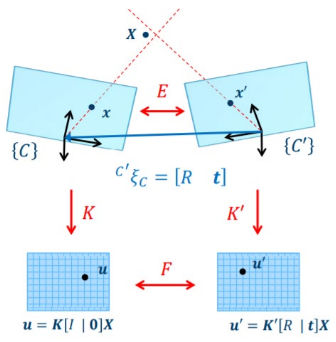
Once the 8 points are obtained, the RANSAC function is rerun till the optimal fundamental matrix is obtained. Our RANSAC function takes a list of both the corresponding keypoints matches and it returns the best inlier points and the optimal Fundamental matrix.

Calculating Essential Matrix: the essential matrix is (3x3) matrix which is simply calculated by E= k' *F* k where k is the camera intrinsic matrix

In the calibrated case, we can estimate and represent the epipolar geometry in terms of the essential matrix E

By estimation, E will be a homogeneous matrix only restricted by xEx'=0

But we also know that we can construct E non homogeneously as $E = \boldsymbol{t} \times R$



In 1981 H. C Longuet-Higgins proved that one could recover the relative pose $\xi C\ C' = R\ \boldsymbol{t}$ from the essential matrix up to the scale of $\boldsymbol{t}$. He argued that, up to the scale of $\boldsymbol{t}$, there are 4 theoretical solutions, but only 1 for which the scene points would be in front of both cameras – This additional constraint has later been named the cheirality constraint

Since we only can estimate E up to scale, we can always rescale it so that the SVD of E has the form:

$$E = UDV^T = \begin{bmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 & \boldsymbol{u}_3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1^T \\ \boldsymbol{v}_2^T \\ \boldsymbol{v}_3^T \end{bmatrix}$$

where $det(\mathrm{U}) = det(V) = 1$

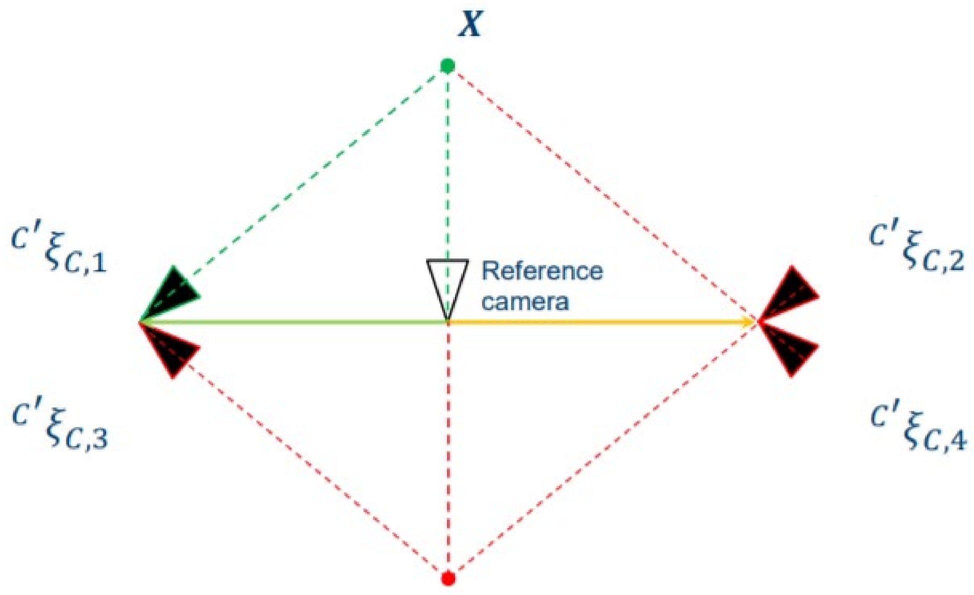
Then one can show that

$R \in \{U\mathrm{W}V^{\mathrm{T}}, UW^{\mathrm{T}}V^{\mathrm{T}}\}$

$\boldsymbol{t} = \pm\lambda\boldsymbol{u}3$; $\lambda \in \mathbb{R}\backslash 0$

Where W is:

$$\mathrm{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



So the 4 candidate poses are:

$$\begin{aligned}
{}^{C'}\xi_{C,1} &= [UWV^T \quad \boldsymbol{u}_3] \\
{}^{C'}\xi_{C,2} &= [UWV^T \quad -\boldsymbol{u}_3] \\
{}^{C'}\xi_{C,3} &= [UW^TV^T \quad \boldsymbol{u}_3] \\
{}^{C'}\xi_{C,4} &= [UW^TV^T \quad -\boldsymbol{u}_3]
\end{aligned}$$

Their relation is shown in the figure for the case when ξ C , 1 C ′ is the correct pose. In general there is no way of knowing the correct candidate without imposing the cheirality constraint. In theory it suffices to triangulate a single scene point X in order to determine the correct pose, but for robustness several points could be checked.

Check For cheirality constraint:

To check this constraint, we need to note this steps:
1) m1 and m2 - First rows of P.
2) m3 and m4 - third and the fourth rows of inv(P).
3) P is generated by taking a pair of rotation and translation
3) Upon getting the X or the point in 3D world, we will find X' wrt to other frame which is X'.
4) If only X and X' are positive, that is the point is front of both cameras. Then we select that points
5) There is error if both the solutions of rotation and translation gives us x and x' as positive.
6) If no x and x' turns out to be positive, we pass identity as the solution.
6) If Z is negative, we negate the entire translation as we want positive z movement.

Afterwards, we do continuous generation of H matrices and column extraction to get camera center. We do it to convert relative rotations and translations w.r.t world. By using the formula stated below

H= H * [R* T*; 0 0 0 1]

Output Comparison (Extra Credits):

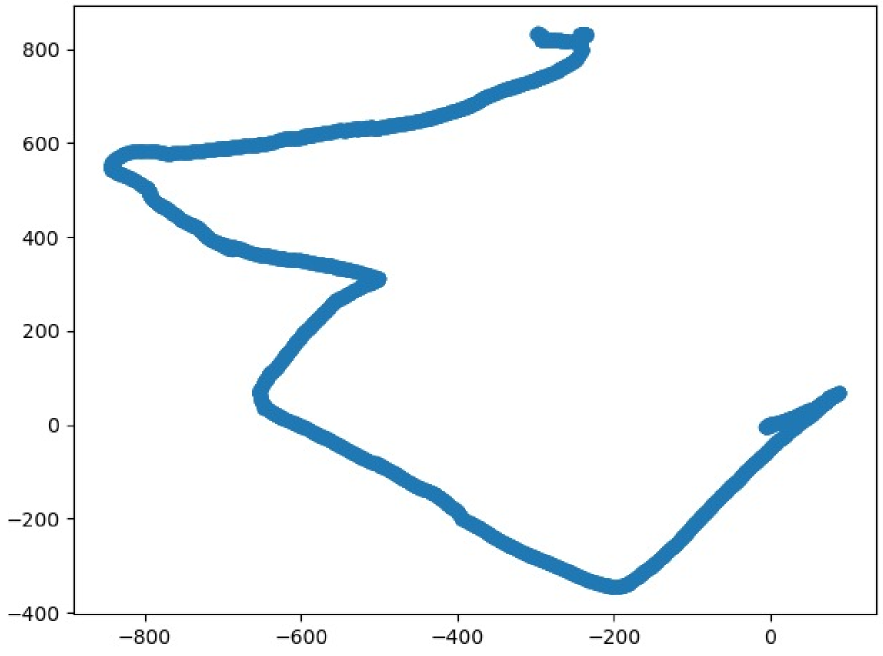


Fig: Output Trajectory obtained using OpenCV functions

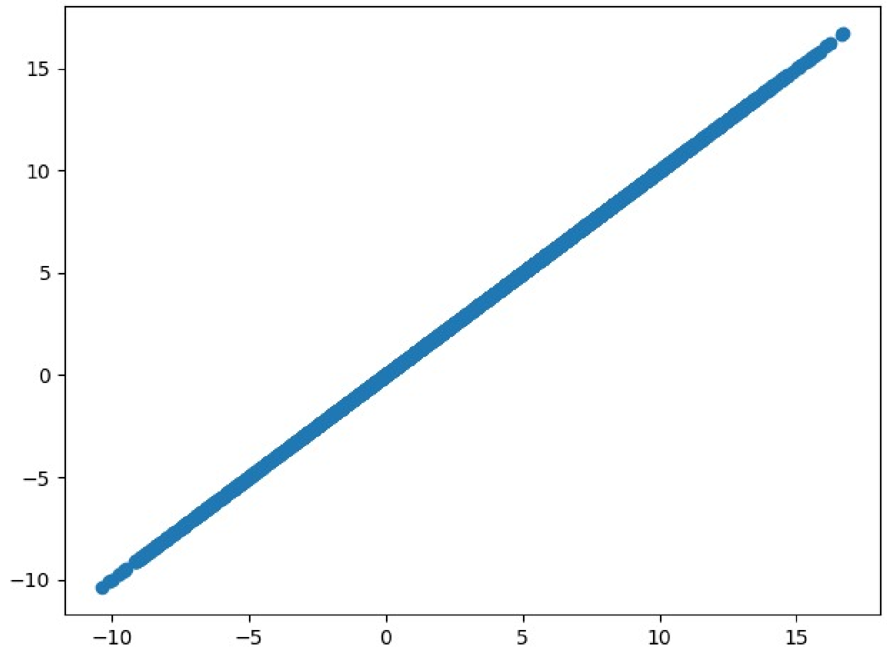


Fig: Output Trajectory obtained using our functions

Non-linear Triangulation (Extra Credits):

Given two camera poses and linearly triangulated points, X, refine the locations of the 3D points that minimizes reprojection error. The linear triangulation minimizes algebraic error. Reprojection error that is geometrically meaningful error is computed by measuring error between measurement and projected 3D point:

$$\underset{\mathbf{X}}{\text{minimize}} \sum_{j=\{1,2\}} \left(u^j - \frac{\mathbf{P}_1^{j\mathsf{T}}\tilde{\mathbf{X}}}{\mathbf{P}_3^{j\mathsf{T}}\tilde{\mathbf{X}}}\right)^2 + \left(v^j - \frac{\mathbf{P}_2^{j\mathsf{T}}\tilde{\mathbf{X}}}{\mathbf{P}_3^{j\mathsf{T}}\tilde{\mathbf{X}}}\right)^2,$$

where j is the index of each camera, Xe is the homogeneous representation of X. PT i is each row of camera projection matrix, P. This minimization is highly nonlinear because of the divisions. The initial guess of the solution, X0, estimated via the linear triangulation is needed to minimize the cost function

Perspective-n-Point (Extra Credits):

Given 3D-2D point correspondences, and linearly estimated camera pose (C, R), we need to refine camera pose (position and orientation), such that the reprojection error is minimized. The linear PnP minimizes algebraic error. Reprojection error that is geometrically meaningful error is computed by measuring error between measurement and projected 3D point:

$$\underset{\mathbf{C},\mathbf{R}}{\text{minimize}} \sum_{j=1}^{J} \left(\left(u_j - \frac{\mathbf{P}_1^{\mathsf{T}}\tilde{\mathbf{X}}_j}{\mathbf{P}_3^{\mathsf{T}}\tilde{\mathbf{X}}_j}\right)^2 + \left(v_j - \frac{\mathbf{P}_2^{\mathsf{T}}\tilde{\mathbf{X}}_j}{\mathbf{P}_3^{\mathsf{T}}\tilde{\mathbf{X}}_j}\right)^2\right),$$

where Xe is the homogeneous representation of X. PT i is each row of camera projection matrix, P which is computed by KR I3×3 −C . A compact representation of the rotation matrix using quaternion is a better choice to enforce orthogonality of the rotation matrix, R = R(q) where q is four dimensional quaternion, i.e

$$\underset{\mathbf{C},\mathbf{q}}{\text{minimize}} \sum_{j=1}^{J} \left(\left(u_j - \frac{\mathbf{P}_1^{\mathsf{T}}\tilde{\mathbf{X}}_j}{\mathbf{P}_3^{\mathsf{T}}\tilde{\mathbf{X}}_j}\right)^2 + \left(v_j - \frac{\mathbf{P}_2^{\mathsf{T}}\tilde{\mathbf{X}}_j}{\mathbf{P}_3^{\mathsf{T}}\tilde{\mathbf{X}}_j}\right)^2\right),.$$

Challenges Faced:

Finding the Fundamental matrix with using the normalized points after passing pre-defined points into our functions

Broadcasting, Dimensionality matching and optimizing the functions errors

Dealing with long run time and memory allocation errors.

Searching for the best optimal feature detection algorithm to detect less and good feature point. Lastly we decided to go with orb as that fulfilled our requirement.

We faced issue in sorting the key points and to solve it we used BF matcher for matching the corresponding descriptors in order to sort them.

References:

1. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_orb/py_orb.html
2. https://cmsc426.github.io/sfm/
3. https://www.uio.no/studier/emner/matnat/its/nedlagte-emner/UNIK4690/v16/forelesninger/lecture_7_3-pose-from-epipolar-geometry.pdf
4. https://www.cis.upenn.edu/~cis580/Spring2015/Projects/proj2/proj2.pdf