Estimating Pose of Quadrotor from Stereo Correspondences

MEAM 620 Project 2, Phase 2

March 28, 2024

1 Introduction

For this project you will be estimating the pose of a flying robot from a set of stereo correspondences implementing the algorithmic ideas we discussed in class. In order to further ground the assignment we are having you apply the algorithms you write to real data acquired by a quadrotor platform. For this we are using portions of the EuRoc dataset collected at ETH Zurich. We are providing a paper describing this dataset and the platform on which it was collected as one of the documents associated with this assignment. I would definitely reccomend taking a quick look at this paper. Note that this time we are providing you with the images as well as the IMU data which makes the dataset much larger.

2 Code Organization

The code packet that you are being provided with as part of this assignment is organized in the usual manner. In the top level directory there is a file entitled `setup.py` which you should run in order to install all of the needed packages. The `proj2_2` package is divided into 3 subdirectories.The `util` directory contains a set of tests you can run on your code via the unit test script `test.py`. The `dataset` directory contains a small portion of the EuRoc dataset which is comprised of stereo and IMU data.

The code directory contains a set of code files and sandbox files which constitute the coding assignment. The file `stereo.py` contains a series of utilities that we are providing for reading information from the dataset, you should not modify this file. The file `estimate_pose_ranscac.py` contains the skeleton of an implementation of the RANSAC based stereo odometry system described in the lectures.

The remaining sandbox scripts are codes that we provide for testing your implementation. The script `sandbox_stereo.py` reads a pair of stereo images from the dataset and plots some figures to show you the outputs. You should be able to run this script successfully after you have run the setup script, if you get errors it means that something isn't configured correctly. You should feel free to edit this script to run the analysis on other stereo pairs in the dataset. The `sandbox_estimate_pose.py` script runs your completed RANSAC implementation against the provided stereo dataset to estimate the pose of the aircraft. After it runs it produces a plot showing the estimated position and orientation over time along with a plot showing the inliers and outliers for each stereo pair processed.

3 Task 1: Least Squares Solve

In order to complete the RANSAC implementation provided in `estimate_pose_ransac.py` you will need to implement a function which estimates the rotation and translation from a set of stereo correspondences. In this case the function you are asked to write, solve_w_t, which takes as input two $3 \times n$ arrays, uvd1 and uvd2, indicating the n corresponding stereo measurements in frames 1 and 2 respectively. The first row of each array contains the normalized coordinate $u' = (X/Z)$, the second row the normalized coordinate $v' = (Y/Z)$

and the last row the normalized disparity $d' = (1/Z)$. The last parameter is a rotation object corresponding to the initial base rotation R_0. Your function should return two 3×1 vectors corresponding to the recovered w and t estimates. You may find it useful to consider the numpy library function lstsq which computes a least squares estimate for a linear system.

4 Task 2: Finding inliers

Your previous function is designed to compute estimates for rotation and translation without considering the possibility of outliers. As part of the RANSAC procedure we are required to identify inlier sets to a proposed solution and this is what your function find_inliers is supposed to do. This function should take as input the two 3×1 vectors corresponding to the proposed w and t values, the aforementioned matrices uvd1 and uvd2 containing the normalized stereo measurements, the initial base rotation R_0 and a threshold value which is used to decide which correspondences are inliers and which are not.

To decide which correspondences are inliers you will need to compute the discrepancy vector δ, which is described in the lecture slides, for each of the correspondences in turn. Correspondences for which $\|\delta\|$ is less than the threshold value are considered inliers otherwise they are considered outliers. Your routine should produce as output a boolean array with n entries, one for each stereo correspondence. Correspondences which are inliers should be indicated with a True value in the output array and those which are outliers should be indicated with a False entry.

5 Task 3: Implementing RANSAC

Once you have implemented the previous two functions you should be in a position to finish the implementation of the function ransac_pose which implements the RANSAC algorithm on the available sensor data. The first two inputs to the function are two $3 \times n$ arrays indicating the n corresponding stereo measurements in frames 1 and 2 respectively. The first row of each array contains the normalized coordinate $u' = (X/Z)$, the second row the normalized coordinate $v' = (Y/Z)$ and the last row the normalized disparity $d' = (1/Z)$. Note that these two arrays have the same structure as the ones passed to solve_w_t but they can contain outliers where solve_w_t blithely assumes that all of the correspondences are inliers. The third parameter is a rotation object corresponding to the initial base rotation R_0. The remaining two parameters ransac_iterations and ransac_threshold indicate the number of RANSAC iterations to be performed and the threshold to be used to decide inliers respectively. Note that the last parameter is the same value that you will pass to the find_inliers function. When the ransac_iterations parameter is zero all of the correspondences should be treated as inliers and passed to the solve_w_t function.

Your function should return two 3×1 vectors corresponding to the recovered w and t estimates along with a boolean array corresponding to the output of the find_inliers function for the final solution.

Hint: You may find the function np.random.choice useful to select random subsets of correspondences. You may also find it useful to use logical indexing to select subsets of correspondences from the input arrays.

6 Slide Elements

We would like you to include a single slide with your final submission that contains the following elements:

1. Please include both of the plots that the sandbox code produced when you ran it.
2. Please include plots showing what happens when you rerun the code with the ransac_iterations parameter set to zero
3. Comment on any differences in the results between the two runs

4. The following pair of equations was presented in the course slides. Please fill in the missing steps showing how the second equation is derived from the first.

$$\delta = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \left[\left[R \begin{pmatrix} u_2' \\ v_2' \\ 1 \end{pmatrix} + d_2' T \right] - \left(\frac{Z_1}{Z_2} \right) \begin{pmatrix} u_1' \\ v_1' \\ 1 \end{pmatrix} \right]$$

$$\delta = \begin{pmatrix} 1 & 0 & -u_1' \\ 0 & 1 & -v_1' \end{pmatrix} \left[R \begin{pmatrix} u_2' \\ v_2' \\ 1 \end{pmatrix} + d_2' T \right]$$

5. Any other details we should be aware of (collaborations, references, etc.) if any.

7 Submission

When you are finished, submit your code via Gradescope which can be accessed via the course Canvas page. Your submission should contain:

1. The `estimate_pose_ransac.py` as well as any other Python files you need to run your code
2. Your slide

Shortly after submitting you should receive an e-mail from `no-reply@gradescope.com` stating that your submission has been received. Once the automated tests finish running the results will be available on the Gradescope submission page. There is no limit on the number of times you can submit your code.

Please do not attempt to determine what the automated tests are or otherwise try to game the automated test system. Any attempt to do so will be considered cheating, resulting in a 0 on this assignment and possible disciplinary action.