MEAM 620 Project 2 Phase 2

Due: Thursday, April 12, 2018 at 11:59pm

1 Overview

In this phase, you will implement a vision-based 3-D velocity estimator and compare against the ground truth provided by the Vicon. You will use the same dataset as Project 2, Phase 1. All calibration parameters remain the same. The following sections give you a logical flow of the estimation procedure.

2 Corner Extraction and Tracking

You will first need to extract corners in each image. You may choose to use MATLAB's built in corner detectors [2, 3], external MATLAB-based computer libraries [4], find implementations on the Internet, or write your own corner detector. You are safe to assume that all detected corners will be on the ground plane. Note that the extracted corners should include not only the AprilTag [1] corners, but also corners from the random scribble.

For all corners in the image, you will then compute the sparse optical flow between two consecutive images using the KLT tracker [5]. You may again choose between [2, 3, 4], or find the tracker implementation on the Internet, or write your own implementation. This sparse optical flow, together with the time stamp of the image, will give you $[\dot{x}, \dot{y}]^T$ in the calibrated image frame. Note that the timestamps can have a bit of jitter, which adds noise to the time measurement. You can assume that the images were taken approximately at a constant rate and simply use a low-pass filter on the Δt to get better results.

3 Velocity Estimation

Given a corner in the calibrated image frame $[x, y]^T$ and its optical flow $[\dot{x}, \dot{y}]^T$, the following linear equation holds:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1(x, y, Z) \\ \mathbf{f}_2(x, y, Z) \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_z \\ \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix} \tag{1}$$

where $[V_x, V_y, V_z, \Omega_x, \Omega_y, \Omega_z]^T$ are the linear and angular velocities to be estimated. Both the functions $\mathbf{f}_1(.)$ and $\mathbf{f}_2(.)$ return 1×6 vectors. Z is the depth of the corner. Assuming that all corners are on the floor, Z can be computed based on the estimated height and rotation of the quadrotor using the AprilTags (part of the results from Project 2, Phase 1). Note that Z does not necessarily equal the height of the quadrotor due to nonzero roll and pitch.

4 RANSAC-Based Outlier Rejection

It is likely that there are outliers in the optical flow computation. You will need to reject outliers using RANSAC. Three sets of constraints are required to solve the system above (1), and thus a 3-point RANSAC

can be used for outlier rejection. It is recommended to recompute the velocities using (1) with all inliers.

5 Compare Against Ground Truth

As in Phase 1, the Vicon data is present as two variables, `time` and `vicon`. The `time` variable contains the timestamp for the Vicon data while the `vicon` variable contains the actual data in the following format:

$$\begin{bmatrix} x & y & z & \text{roll} & \text{pitch} & \text{yaw} & v_x & v_y & v_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^T$$

Note that the optical flow-based velocity estimation will be in the frame of the camera, while the Vicon velocity is given in the world frame. You will need to transform the coordinate frames between the camera, the quadrotor, and the world to properly compare your velocity estimates against the ground truth. The Vicon velocity estimate can be noisy at some point, but you should be able to get a reasonable ground truth during majority of the flight time.

6 Submission

As always, we'll use the `turnin` system for submission. The project name for this phase is called `proj2phase2`, so the command you'll use for `turnin` would be

```
$ turnin -c meam620 -p proj2phase2 -v *
```

Your submission should include:

- A `README` file containing anything specific we should be aware of.
- Files `init_script.m`, `estimate_vel.m` and any additional files your code requires to run.

Shortly after submitting using `turnin` you should receive an email from `meam620@seas.upenn.edu` stating that your submission has been received. After that, you can check the status of your submission on the monitor webpage at https://alliance.seas.upenn.edu/~meam620/monitor/.

Once the automated tests finish for your code, you should receive an email containing your test results. This email will contain plots of your estimates compared to the ground truth and also tell you how long each iteration of your code takes. You should write your own functions for evaluating the error between your estimate using vision and the ground truth. Your grade would depend on the time taken per iteration and the error between your estimate compared to ground truth.

References

[1] April Tags, http://april.eecs.umich.edu/wiki/index.php

[2] MATLAB Image Processing Toolbox, http://www.mathworks.com/products/image/

[3] MATLAB Computer Vision System Toolbox, http://www.mathworks.com/products/computer-vision/

[4] VLFeat, http://www.vlfeat.org/

[5] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in Proc. of the Intl. Joint Conf. on Artificial Intelligence, Vancouver, Canada, Aug. 1981, pp. 24-28.