CSC2503—Foundations of Computer Vision, Fall 2017

Assignment 2: Fundamental Matrix & Homography Estimation

Due: 2pm, Wednesday, November 1

This assignment explores (a) the estimation of the fundamental matrix from two perspective views of a general (non-planar) 3D scene and (b) the estimation of 2D homographies from two perspective views of a planar surface in 3D.

To begin: In the handout code we provide a script file `grappleFmatrix.m` which reads point correspondence data, and uses the RANSAC algorithm to estimate a fundamental matrix (i.e. an F-matrix). This script file uses several M-files in the `./utils/` directory, and data from the `./data/` directory. Familiarize yourself with this code.

What to submit. (1) Write a short report addressing each of the itemized questions below. Your report should be in PDF format and should include the images requested. LaTeX- or Word-formatted reports are preferred; a hand-written derivation that has been scanned to PDF is also fine *as long as it is very legible*. Assume the marker knows the context of all questions, so do not spend time repeating material in the hand-out or the class notes. (2) Any Matlab file you created or altered. We strongly prefer not to have to look through your code. We only want it to chase down things which are not clear from your report.

How to submit. Create a tarfile called `assign2.tar.gz` and use the following command on a Teaching Labs (a.k.a CDF) computer to submit it electronically:

```
> submit -c csc2503h -a assign2 assign2.tar.gz
```

Late policy (revised!) There will be a 10% marks deduction for each day late, for up to three days. Deductions begin immediately after the due date and time (ie. 2pm on November 1st). Assignments submitted more than 72 hours late will not be accepted.

Part A: Fundamental Matrix Estimation

The script file `dinoTestF.m` uses exact point correspondences obtained from `utils/projectDino.m` for two perspective views of points on a single rigid object (i.e., Dino). It then computes the F-matrix from these corresponding points. Here we wish to examine the sensitivity of the F-matrix estimation to: i) noise in the estimation of the image point positions, ii) the non-planar depth variation in the scene, and iii) changes in the viewing geometry.

1. Your first task is to compute a ground-truth fundamental matrix, say F_0, against which we can compare our estimators below. Toward this end, from the calibration matrices M_{int} and M_{ext} provided by `projectDino.m` for each of the two views used in `dinoTestF.m`, compute the F-matrix F_0 directly from these calibration matrices. Your write-up should show the derivation for computing F_0 from this information.

2. We wish to compare the F-matrix, say F_1, computed in the original hand-out code `dinoTestF.m`, with the ground-truth matrix F_0 computed in (1) above. In order to compare two F-matrices, choose a regularly spaced grid of points in one image. Then, for each point in this grid, compute the epipolar lines in the second image using each of the two F-matrices, F_0 and F_1. Crop the epipolar line from matrix F_0 to be within the image region (see `utils/cropLineInBox.m`). Ignore

any line which does not intersect the image region at all (i.e. if `utils/cropLineInBox.m` returns a NaN). Next compute the maximum perpendicular distance from this cropped epipolar line to the corresponding (uncropped) epipolar line for F_1. Finally compute the maximum, taken over all pairs of lines generated by F_0 and F_1 applied to the grid points, of these maximum perpendicular distances. This maximum of maximum perpendicualr errors will be used as the error measure between F_0 and F_1. Note this error measure is not symmetric in the roles of F_0 and F_1.

Using this error measure, what is the error between the F-matrix computed by the handout code `dinoTestF.m` and the one you computed in part (1) above?

3. Here we consider the effect of image position errors in the positions of the individual points. We will simulate this noise by adding Gaussian random noise, with standard deviation σ_n and mean zero, to the first and second components in each of the corresponding points $\{\vec{p}_k\}_{k=1}^K$ and $\{\vec{q}_k\}_{k=1}^K$ provided by `projectDino.m`. (Do not perturb the third coordinate, with value 1, in these points.) Estimate the F-matrix using the M-file `linEstF.m` given only these noisy input points. Compute the error between this estimated F-matrix and the true F-matrix—see parts (1) and (2).

   Repeat this process several times with the same σ_n, but different noise samples, in order to get an estimate for the median value of the error between the computed F-matrix and the true F-matrix. Plot this median error as a function of σ_n. Ideally, the estimate of F should have an error that is roughly proportional to σ_n for small values of σ_n. For larger values the error should increase rapidly, up to a point where the error is a signicant fraction of the size of the image. At this point our estimate of F has broken down.

   Comment on the results. For example, is there a maximum value of σ_n for which you can get reasonably small errors in the recovered F-matrix?

4. Do the same as in part (3) above, only set the argument `NUM_RESCALE` for `linEstF.m` to be 0, so Hartley's normalization is not used. Comment on the difference in the stability of the estimated F-matrices with and without normalization. (After completing this question, set `NUM_RESCALE` to be true again for the remaining questions!)

5. The parameter `sclZ` in `projectDino` scales the Z component of Dino (i.e. the side-to-side width of the animal) in its original coordinate frame. A small value of `sclZ`, eg. `sclZ=0.1`, corresponds to a flattened Dino. Dino is perfectly planar when `sclZ=0`.

   Do the same as part (3) above, only use `sclZ=0.1` instead of its default value of 1. Does the true F-matrix change from the one for the case `sclZ=1.0`? Explain. (Beware, if you do not give a rotation matrix as an argument to `projectDino`, then it will rotate the two cameras whenever `sclZ` is changed.) Is the estimation of F more tolerant to image position noise (i.e. the added noise described in part 3) for this flattened case than the original case considered in part (3)?

6. The parameter d in projectDino specifies the position of the camera's nodal point in a coordinate frame centered on Dino itself. In the hand-out code the two cameras have been symmetrically placed about Dino's Z-axis, with `d` set to $(50, 0, -150)^T$ and $(-50, 0, -150)^T$ for the two cameras. The rotation of each camera is computed in `projectDino` so that the mean of Dino's vertices is mapped to the center of the image.

   Plot the median error in F versus σ_n as in part 3 above, only use a smaller separation between the cameras, say `d` $= (5, 0, -150)^T$ and $(-5, 0, -150)^T$. Use `projectDino` to compute the camera rotations such that the cameras are directed towards the mean point on Dino. (Beware the true F-matrix will change, due to this change in the positions of the centers of projection and the change in these camera rotations (see parts (1) and (2).) Is the estimation of F more tolerant

to image position noise when the cameras are closer together than the original case considered in part (3)?

Part B: Homography Estimation

Let $\{(\vec{p}_k, \vec{q}_k)\}_{k=1}^K$ be the homogeneous pixel coordinates for corresponding points for two perspective views of a single planar surface. In particular, $\vec{p}_k = (p_{k,1}, p_{k,2}, 1)^T$ where $(p_{k,1}, p_{k,2})^T$ is the (x, y) pixel location of the k^{th} point in the first image. Similarly, $\vec{q}_k = (q_{k,1}, q_{k,2}, 1)^T$, where $(q_{k,1}, q_{k,2})^T$ is the (x, y) pixel location of the corresponding point in the second image. Since these corresponding points are all from a single planar surface, they must be related by a 2D homography. That is

$$\alpha_k \vec{q}_k \;=\; H\vec{p}_k, \text{ for } k = 1, \ldots, K, \tag{1}$$

where H is a 3×3 matrix. The scalar $\alpha_k > 0$ above accounts for the normalization of the third coordinate of $\vec{q}_k$, which we take to be 1. (Often equation (1) is written in homogeneous image coordinates, in which case $\vec{q}_k$ need not be normalized, and hence the scalar k can be dropped. In this case, the vector $\vec{q}_k$ would be normalized to have a third coordinate equal to 1 only when we wish to work out which pixel coordinates it refers to.)

Given these observed corresponding points $\{(\vec{p}_k, \vec{q}_k)\}_{k=1}^K$, we wish to estimate the homography H. To do this, first note that we can eliminate the scalar α_k by observing that equation (1) states that $\vec{p}_k$ and $H\vec{q}_k$ are in the same direction in 3D, and therefore their cross product must be zero. That is,

$$\vec{q}_k \times (H\vec{p}_k) = \vec{0} \text{ for } k = 1, \ldots, K. \tag{2}$$

This gives three linear equations for H for each k. Only two of these three equations are linearly independent. In fact, we can use just the first two rows of (2) for each k in order to estimate H.

1. Complete the M-file `linEstH.m` so that, given a set of corresponding points $\{(\vec{p}_k, \vec{q}_k)\}_{k=1}^K$, it solves the least squares problem

$$\arg\min_H \sum_{k=1}^{K} \sum_{j=1}^{2} [\; \vec{q}_k \times (H\vec{p}_k) \;]_j^2 \,. \tag{3}$$

   Here $[\vec{v}]_j$ denotes the j^{th} coordinate of the vector $\vec{v}$.

   Your solution must comply with the description of both the input and output in the header of the current M-file `linEstH.m` in the handout code (see directory `./utils`). The file begins with the same normalization of $\vec{p}_k$ and $\vec{q}_k$ as in `linEstF.m`. Besides the header, you therefore only need to modify the code below the line that reads

```
%% Make constraint matrix A.
```

   In particular, after normalizing the input points $\vec{p}_k$ and $\vec{q}_k$, solve the least squares problem (3) written in terms of these normalized points. The solution is the 3×3 homography matrix H_0. Then transform H_0 back to the original coordinates (beware, this transformation is not identical to the one used in `linEstF.m`). Finally, since H is only determined up to a scalar multiple, rescale H so that the sum of the squares of its 9 elements is one.

   In the written portion of your answer for this question include a derivation of the equations you solved for H_0, and how H_0 gets transformed back to the desired H matrix.

2. In order to deal with outliers in the correspondence set $\{(\vec{p}_k, \vec{q}_k)\}_{k=1}^{K}$ we will use RANSAC to identify a set of inliers. An implementation of RANSAC for the purpose of computing the fundamental matrix (aka the F-matrix) is provided in `grappleFmatrix.m` in the handout code. Write a script file `grapple2DHomog.m`, which is a modified copy of `grappleFmatrix.m`, so that it estimates a 2D homography H from the correspondence data instead. Replace the output of epipolar lines in `grappleFmatrix.m` with code that outputs the two input images, with one of these images warped by the computed homography. (Use the M-file `homogWarp.m` provided in the handout code to compute the warped image.)

   Show the results of running `grapple2DHomog.m` on the Wadham college correspondence data provided with the handout code.

Academic Honesty. Cheating on assignments has very serious repercussions for the students involved, far beyond simply getting a zero on their assignment. This is especially so for graduate students.

This assignment is **strictly individual work.** It should not be discussed with anyone, even at the level of ideas. The TAs are very experienced in detecting signs of collaboration between students and there is a **zero tolerance on cheating**. Even minor infractions will be identified and reported to the department and the Dean's office.

The course's assignments cover widely-used vision techniques, some of which have been used in prior instalments of this course. Answers to some of the written questions—and even code for some of the assignments—may be just a mouse click (or google search) away. You must resist the temptation to search for, download and/or adapt someone else's solutions and submit them as your own *without proper attribution.* Accidentally stumbling upon a solution is no excuse either: the minute you suspect a webpage describes even a partial solution to this assignment, either stop reading it or cite it in the report you submit. Simply put: if an existing solution is easy for you to find, it is just as easy for us to find it as well. In fact, *you can be sure we already know about it!!*

Clearly, web searches related to generic Matlab commands and/or generic concepts from calculus or probability theory are not out of bounds. However, if you have any doubt about whether or not you should cite material that you read online (or somewhere else) as you were working on the assignment, the safest thing to do is to just *cite it.* In such a case, your grade will depend on how much of the solution is your own work and how much of it came from the cited sources. But regardless of the grade you get, you will not be in violation of the University's Academic Integrity policies.