Computer Vision HW 4

July 23, 2019

Background: Planar Homography

Suppose we have two cameras C_1 and C_2 looking at a common plane Π in 3D space. Any 3D point P on Π generates a projected 2D point located at $p \equiv (u_1, v_1, 1)^T$ on the first camera C_1 and $q \equiv (u_2, v_2, 1)^T$ on the second camera C_2. Since P is confined to the plane Π, we expect that there is a relationship between p and q. In particular, there exists a common 3×3 matrix H, so that for any p and q, the following condition holds:

$$p \equiv Hq,$$

where the equality $\equiv$ means p is proportional to Hq. We call this relationship *planar homography*. It turns out this relationship is also true for cameras that are related by pure rotation without the planar constraint.

Given a set of points $\mathbf{p} = \{p_1, p_2, \ldots, p_N\}$ in an image taken by camera C_1 and corresponding points $\mathbf{q} = \{q_1, q_2, \ldots, q_n\}$ in an image taken by C_2, there is a set of $2N$ independent linear equations in the form

$$Ah = 0$$

where h is a vector of the elements of H and A is a matrix composed of elements derived from the point coordinates. In particular, if

$$h^T = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} & h_{33} \end{bmatrix},$$

then every pair of points contribute with two equations, namely with

$$A = \begin{bmatrix} u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1x_1 & -v_1x_1 & -x_1 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -u_1y_1 & -v_1y_1 & -y_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2x_2 & -v_2x_2 & -x_2 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -u_2y_2 & -v_2y_2 & -y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_N & v_N & 1 & 0 & 0 & 0 & -u_Nx_N & -v_Nx_N & -x_N \\ 0 & 0 & 0 & u_N & v_N & 1 & -u_Ny_N & -v_Ny_N & -y_N \end{bmatrix}.$$

We need to estimate the H that minimizes the homogeneous linear least squares system. Since the minimum error can be given by

$$e = \min_{h^Th=1} (h^T A^T A h) = \min_{h^Th=1} (\lambda h^T h) = \lambda,$$

the solution to this problem is the eigenvector to A^TA with the smallest eigenvalue λ. You will need this for Part 1.

1 Homogeneous Least Square Implementation (20 pts)

Now we will implement the algorithm to find H mathematically. Your job is to implement the function

```
H2to1 = computeH(p1, p2)
```

Inputs: `p1` and `p2` should be $N \times 2$ matrices of corresponding (x, y) coordinates between two images.

Outputs: `H2to1` should be a 3×3 matrix encoding the homography that best matches the linear equation derived above for Equation 8 (in the least squares sense). *Hint*: Remember that a homography is only determined up to scale. The functions `numpy.linalg.eig()` or `numpy.linalg.svd()` will be useful. Note that this function can be written without an explicit for-loop over the data points.

2 RANSAC Implementation (40 pts)

Partial Credits

- Find model for randomly selected points (10 pts)
- Extend model to all inliers of model (15 pts)
- Iterate correctly to get best-fitting H (15 pts)

Description Note that the least squares method you implemented for computing homographies is not robust to outliers. When correspondences are determined automatically, some mismatches in a set of point correspondences are almost certain. RANSAC (Random Sample Consensus) can be used to fit models robustly in the presence of outliers.

Write a function that uses RANSAC to compute homographies automatically between two images:

`bestH = ransacH(matches, locs1, locs2, nIter, tol)`.

The inputs and outputs of this function should be as follows:

Inputs: `locs1` and `locs2` are matrices specifying point locations in each of the images and `matches` is a matrix specifying matches between these two sets of point locations. These matrices are formatted identically to the output of the provided `briefMatch` function.

Algorithm Input Parameters: `n_iter` is the number of iterations to run RANSAC for, `tol` is the tolerance value for considering a point to be an inlier. Define your function so that these two parameters have reasonable default values.

Outputs: `bestH` should be the homography model with the most inliers found during RANSAC.

Pseudocode

```
iterations = 0
bestFit = nul
bestErr = something really large
while iterations < k {
    maybeInliers = n randomly selected values from data
    maybeModel = model parameters fitted to maybeInliers
    alsoInliers = empty set
    for every point in data not in maybeInliers {
        if point fits maybeModel with an error smaller than t
             add point to alsoInliers
    }
    if the number of elements in alsoInliers is > d {
        % this implies that we may have found a good model
        % now test how good it is
        betterModel = model parameters fitted to all points in maybeInliers and alsoInliers
        thisErr = a measure of how well betterModel fits these points
        if thisErr < bestErr {
            bestFit = betterModel
            bestErr = thisErr
        }
    }
    increment iterations
}
return bestFit
```