CMPE 537 Computer Vision Homework-2 Report

Image Stitching

Giray Eryılmaz

Explanation of required parts:

1. **Select Common Points on Images:**
   Done by hand, using ginput.
2. **Homography Estimation:**
   As requested the homography transformation matrix between images is calculated with computeH function. The procedure is as follows:
   - Points are correspondence points provided by the user,
   - They are transformed into homogenous coordinates by adding a 1 as an extra dimension.
   - They are normalized by getting multiplied by the normalization transformation which is calculated by my 'get_normalization_matrix' function.
     The transformation is formulated as such:

(i) Transform the image coordinates according to transformations $\tilde{\mathbf{x}}_i = \mathrm{T}\mathbf{x}_i$ and $\tilde{\mathbf{x}}'_i = \mathrm{T}'\mathbf{x}'_i$.

(ii) Find the transformation $\widetilde{\mathrm{H}}$ from the correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$.

(iii) Set $\mathrm{H} = \mathrm{T}'^{-1}\widetilde{\mathrm{H}}\mathrm{T}$.

Source: Multiple View Geometry in Computer Vision (Hartley and Zisserman)

   - Finally H is calculated according to the formula from the lecture slides.

$$\begin{bmatrix} x_i k_i' & y_i k_i' & k_i k_i' & 0 & 0 & 0 & -x_i x_i' & -y_i x_i' & -k_i x_i' \\ 0 & 0 & 0 & x_i k_i' & y_i k_i' & k_i k_i' & -x_i y_i' & -y_i y_i' & -k_i y_i' \end{bmatrix} \hat{\mathbf{h}} = \mathbf{0}$$

   This should hold for all corresponding point couples. And by using only 4 such correspondences we should be able to find the H because each couple brings 2 equations and we have 9 unknowns with **8** number of freedom. But since the data is noisy we can not algebraically find the solution. Instead we find the transformation that minimizes the error. This can be done via singular value decomposition if you have many points. It works better if you have better (less noisy) or more points.
   You simply stack-up these relations to get matrix A and the you take the SVD of A. The H is the singular vector corresponding to the smallest singular value. On whose direction the least amount of change lies. (Credit goes to lecture slides and of course to the instructor).

3. **Warping**
   As requested I implemented the warp function with designated inputs. It does a backward mapping as requested. The backward mapping works as follows:
   Instead of transforming all points from the previous image to the morphed image, I first determine the shape of the output by just forward mapping the corners of the image then I traverse all points in the output dimension and by doing a backward transform on them

(using H inverse) to get where these points are supposed to be coming from. Of course usually non-integer values are encountered, for interpolation I chose the nearest grid point. It is fairly simple I implemented it myself.

4. **Blending images:**
   I tried both overriding pixeles with the middle image having the highest priority and picking the point with the highest intensity. I believe picking the highest intensity is problematic because, for instance white windows appear twice and look considerably ugly but this is just my personal opinion. On the other hand overriding makes transitions stand out in a bad way. Unfortunately I could not devise smarter methods because of time restrictions.

5. **Experiments:**
   5.1 Number of points:

5 corresponding points: I believe I have picked the points very well.

12 corresponding points: This was supposed to be better than 5 points yet my test with 5 points looks exceptionally fine.

5.2 Point selection:

12 points 3 of them are wrong - no normalization

12 points of which 3 are wrong matches: This is worse 12 correct corresponding points one. But there is always the question of “how wrong?” remains. I choose points somewhat far away from the original position. If you choose just arbitrary points results may be drastically different.

12 points with 5 of them being wrong: This one is worse than 3 wrong points one but not that different.

5.3 Noisy points:

Noisy point with standard deviation = 1, not much affected by the noise

Noisy point with standard deviation = 16, effects of the noise is visible.

Noisy point with standard deviation = 16, no normalization.

Comments on normalization: The idea behind normalization is that the points with small values such as x=10,y=5 should have the same effect with the ones with the high values like x=800,y=900. If we don’t normalize this won’t hold. The larger values will have larger effects.

5.4 Panorama

First I merged l1 - m - r1 and saved the result. Then I merged l2 - the result - r2.