Computing Homography: **Programming Guideline**

Overview

The main part of the homework is to calculate homography transformation, maping from one viewpoint to another. We will formulate it as a least squares estimation problem.

For one pair of correspondent points $\begin{bmatrix} x & y & 1 \end{bmatrix}^T$ and $\begin{bmatrix} x' & y' & 1 \end{bmatrix}^T$ in two different views, the transformation from the first view to the other is given by:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \propto \mathbf{Hx} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\Rightarrow x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \text{ and } y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

And these can be written as equations for parameters of $\mathbf{H}$ (given $h_{33} = 1$):

$$\begin{cases} h_{11}x + h_{12}y + h_{13} - h_{31}xx' - h_{32}yx' = x' \\ h_{21}x + h_{22}y + h_{23} - h_{31}xy' - h_{32}yy' = y' \end{cases}$$

Or in matrix form:

$$A\mathbf{h} = \mathbf{b}$$

$$A = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -xx' & -yx' \\ 0 & 0 & 0 & x & y & 1 & -xy' & -yy' \end{bmatrix}, \mathbf{b} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

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

Similarly, with n pairs of correspondences, we can create a system of $2n$ equations for $\mathbf{h}$: $A\mathbf{h} = \mathbf{b}$, where every two rows of A and $\mathbf{b}$ will represent the relation of each pair. A will be a matrix of size $2n \times 8$ and $\mathbf{b}$ will be a vector of size $2n \times 1$. In order to solve for $\mathbf{h}$, we will need at least 4 pairs. For computational stability, we often want to collect more than 4 pairs and have an overdetermined system where the number of equations is greater than the number of variables ($2n > 8$). In this case, $\mathbf{h}$ can be found by minimizing the squared error $||A\mathbf{h} - \mathbf{b}||_2^2$, $\mathbf{h}$ is then called the least squares solution for the overdetermined system. This might sound complicated, but to find $\mathbf{h}$ in MATLAB, we just need one command $\mathbf{h} = A\backslash\mathbf{b}$.

Implementation

In the homework you are asked to do the following:

1. Identify pairs of correspondent points in two images. At least 4 pairs are needed for the algorithm but it's recommended to use 8 or even more for better results. In this step you might want to explore function **ginput** in MATLAB and might as well **save** those points to a file so that you can reuse later. You can take the following script as a reference, and modify it based on your needs.

```
im1 = imread('left.jpg');
im2 = imread('right.jpg');

imshow(im1)
[x1, y1] = ginput(8);
P1 = [x1, y1]';

imshow(im2)
[x2, y2] = ginput(8);
P2 = [x2, y2]';

save('pairs.mat', 'im1', 'P1', 'im2', 'P2')
```

2. Write a function, *homography.m* for example, to calculate homography transformation from one image to the other using the least squares method. The function should take the corresponding pairs as input and output the transfromation matrix **H**. Your job is basically putting those pairs' coordinates into the matrix A and vector **b** so that MATLAB can do the rest. The function might look something like this.

```
function H = homography(P1, P2)
  % Your code here
  % ..............
  h = A\b;
  h= [h; 1];
  % Be aware of how reshape function works
  % and the maping between vector h and matrix H
  H = reshape(h, 3, 3)';
end
```

3. Finally, using the homography to do the transform from one view to the other. For this, you should look into functions **maketform** and **imtransform** in MATLAB. You might have a *main* script similar to this one.

```
% load what we saved before
load pairs.mat

% transfrom im2 to the same view point as im1
H21 = homography(P1, P2)
% note that MATLAB stores the transformation matrix differently
T21 = maketform('projective', H21');
im21 = imtransform(im2, T21);
figure, imshow(im21)
figure, imshow(im1)
```

4. You have repeat this for 2 pairs of images. For each pair, you need to calculate transformation from one image to the other. So totally you will have 4 transformations to calculate. During the process, remember to save neccessary information needed for your report.