2 Homography

Images are related by a homography, i.e., a projective transformation, if they are captured by simply rotating the camera or they are projections of a planar object. In this assignment, you will write programs to compute the homography $H$ and generate some synthetic images based on the $H$. Fig.3 illustrates an example application in which we post images into planar surfaces in other images.

![Figure 3: We post the first two images into the third one on row one. Images on the second row show the results.](image)

As we learned in the class, a homography is determined by at least 4 corresponding point pairs. Estimating $H$ is a standard least square problem in which we have more equations than the variables. In this problem, we would like to find $h$ which is a flattened $H$ such that $Ah = 0$. The matrix $A$ is constructed using the corresponding points. Each two corresponding points $(x_1, y_1)$ and $(x_2, y_2)$ introduce two rows in $A$ which are

\[
\begin{align*}
a_x &= (-x_1, -y_1, -1, 0, 0, 0, x_2x_1, x_2y_1, x_2) \\
a_y &= (0, 0, -x_1, -y_1, -1, y_2x_1, y_2y_1, y_2)
\end{align*}
\]

We know that $h$ is simply the eigen vector that corresponds to the 0 (the lowest) eigen value of $A^T A$. Or equivalently, we can compute the SVD of $A$ and the last column of $V$ is $h$. The function that computes $H$ is as follows:
function H = compH(x1, y1, x2, y2)
% function compH finds the homography H
% (x1,y1) and (x2,y2) are matching points

ax = [-x1, -y1, -ones(size(x1)), zeros(size(x1)), zeros(size(x1)),...
    zeros(size(x1)), x2.*x1, x2.*y1, x2];
ay = [zeros(size(x1)), zeros(size(x1)),...
    zeros(size(x1)), -x1, -y1, -ones(size(x1)), y2.*x1, y2.*y1, y2];
A = [ax; ay];
[U,S,V] = svd(A);
H = reshape(V(:,9), 3, 3);
H = H';

2.1 Programming [20 points]
Write a Matlab program to measure homography and implement the effect as show in Fig.3. If you can do something more than this, you get extra credits.

2.2 Writing [20 points]
Write a report about your code and some of your results in the HTML format.

3 Triangle Warping Sample Code
Try the following code in Matlab. It warps a user defined triangular region from one image to another. You can use it as a basis for your morphing project. For the homography project, the warping is quite similar but it is from a quadrangle to another quadrangle. The following sample code uses the function inTri.

% Click (3) points alternatively on the first and
% the second blank image. (Change the cat image with
% any image you have).
im1 = imread('cat1.jpg');
im1 = im2double(im1);
figure(1);
imshow(im1);
hold on;
figure(2);
imshow(ones(size(im1)));
hold on;

xx1 = [];
yy1 = [];

xx2 = [];
yy2 = [];

for n = 1 : 3
figure(1);
[x1,y1] = ginput(1);
plot(x1, y1, 'r.', 'markersize', 20);
drawnow;
xx1 = [xx1; x1];
yy1 = [yy1; y1];

figure(2);
[x2,y2] = ginput(1);
plot(x2, y2, 'r.', 'markersize', 20);
drawnow;
xx2 = [xx2; x2];
yy2 = [yy2; y2];
end
figure(1); hold off;
figure(2); hold off;

% Warp the triangle from the first image to the second
W = size(im1,2);
H = size(im1,1);
[x,y] = meshgrid(1:W, 1:H);

[w1,w2,w3,r] = inTrix(x(:), y(:), xx2(1), yy2(1), xx2(2), yy2(2), xx2(3), yy2(3));
mask = reshape(r, H, W);
mask = im2double(mask);

u = w1*xx1(1) + w2.*xx1(2) + w3.*xx1(3);
u(u<1) = 1;
u(u>W) = W;
v = w1*yy1(1) + w2.*yy1(2) + w3.*yy1(3);
v(v<1) = 1;
v(v>H) = H;

u = reshape(u, size(im1,1), size(im1,2));
v = reshape(v, size(im1,1), size(im1,2));
im = interp2(im1(:,:,1), u, v);
R = ones(H, W);
R(mask>0) = im(mask>0);

im = interp2(im1(:,:,2), u, v);
G = ones(H, W);
G(mask>0) = im(mask>0);

im = interp2(im1(:,:,3), u, v);
B = ones(H, W);
B(mask>0) = im(mask>0);

% Show the warped image with the triangle mask
figure(2);
imshw(cat(3, R, G, B).* cat(3, mask, mask, mask));