Computer Vision - HW #4

Anthony Bell
April 19, 2015

> Our Goal for this project is to use homography to map coordinates between images in order to 'stitch' them together.

1. Getting correspondance points

We created a function called `getPoints.m` that uses `ginput()` to retrieve input clicks. When the user presses enter, it shows the user the points and asks if the user wants to accept the points. If they type `no` that the image clears and they can select new points. The function returns the points when the user types `yes`.

2. Calculating the homography matrix

We are using the projective transform, so we are using homogenuous coordinates:

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

The projective transform has 8 degrees of freedom so we can set $h_{22} = 1$, giving us the transformed points:

$$x' = u/w = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + 1} \qquad y' = v/w = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + 1}$$

We end up with a system of equations of the form:

$$\begin{pmatrix} & & & & & \vdots & & \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i x_i' & -y_i x_i' \\ 0 & 0 & 0 & x_i & y_i & 1 & -x_i y_i' & -y_i y_i' \\ & & & & & \vdots & & \end{pmatrix} \begin{pmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \end{pmatrix} = \begin{pmatrix} \vdots \\ x_i' \\ y_i' \\ \vdots \end{pmatrix}$$

$$A \qquad h \quad = \quad b$$

This is a overdetermined system of equations that can be solved by multiplying both sides by the transpose of the matrix A solving the resultant system of equations:

$$\begin{aligned} Ah &= b \\ A^T Ah &= A^T b \\ h &= (A^T A)^+ A^T b \end{aligned}$$

We add 1 for the entry h_{22} and reshape h to 3×3 to get our homography matrix H.

3. Warping between image planes

Now that we have a projective matrix we can transform one image to another's frame and merge them. The code to generate project and image and merge it with a reference image is in `warpImage.m`.

We take the input image, generate a list of all coordinates for the image using `meshgrid` and apply the projective matrix to the coordinates to obtain the list of coordinates that the original image is projected to in order to get it's bounding box. Using the max/max of these coordinates we calculate the needed dimensions for the new image. We then apply an inverse mapping of H on the integer values for the dimensions of the new image to get the pixel values for the new image. Since the projective matrix can output negative values, we move all the coordinate values over to make them positive by subtracting by the min of the x and y dimensions. In order to create the merged image we reuse this same image but

superimpose the reference image, adding the same offset that we used to avoid negative indices. The results are shown in the next section.

4. Applying warping to images

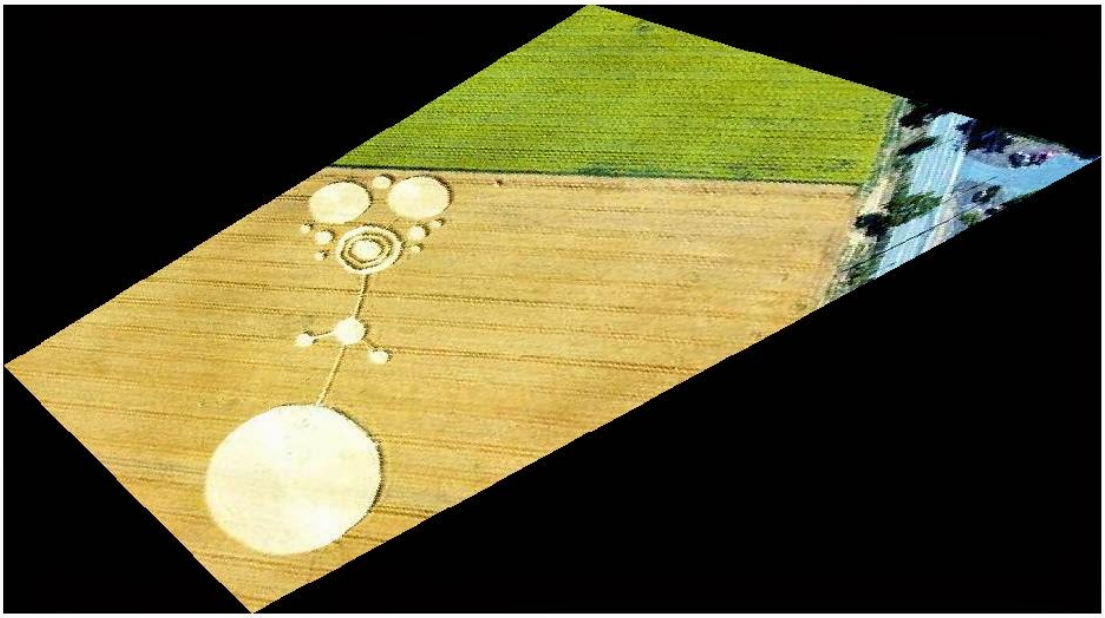
The first images we merge are of a crop circle taken from different views:

First image of the crop circle

Second image of the same crop circle from a different view point

The homography matrix from the first image to the second was calculated for a set of correspondence points. Below is the result of projecting the image and the merging of the projected image with the second image.

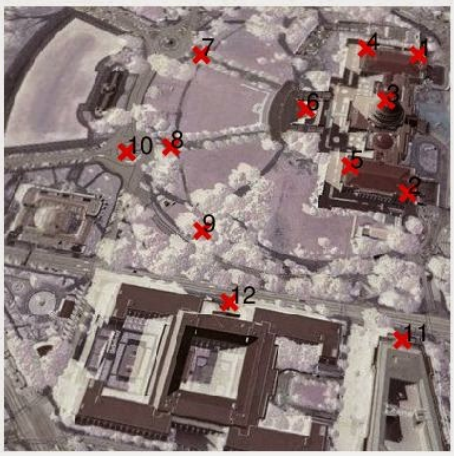
The warped crop circle image.



The merge crop circle images

Next we used aerial images of a capital building. We used a small program to collect correspondence points. the images and the points collected are shown here:

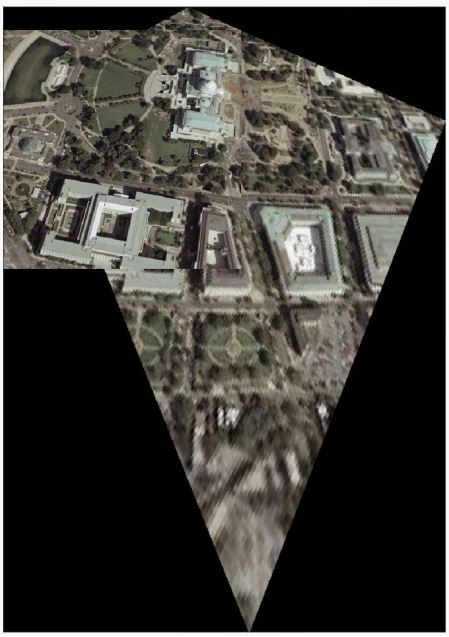
Chosen correspondence points for the building image example



The homography matrix was calculated using the points collected. The warped and merged images are shown below:

The first image of the building projected into the frame of the second

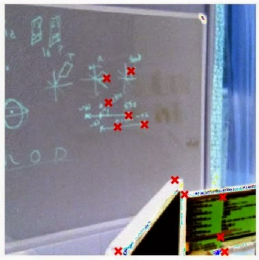
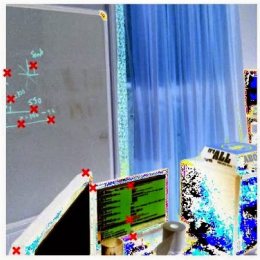
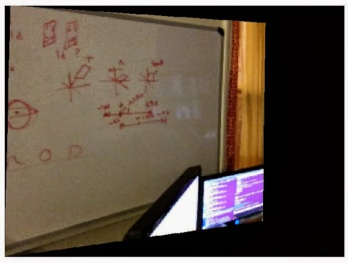
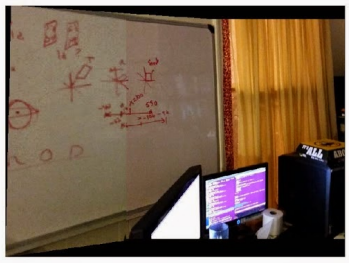
The projection of the first image is stitched to the second.



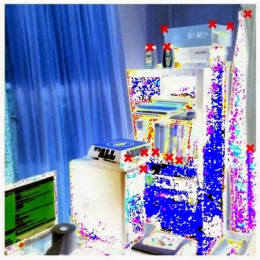
5. Applying warping to pictures taken

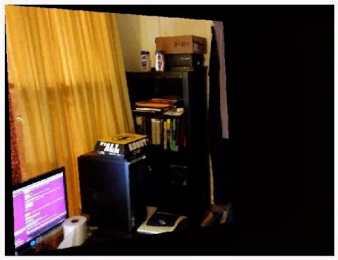
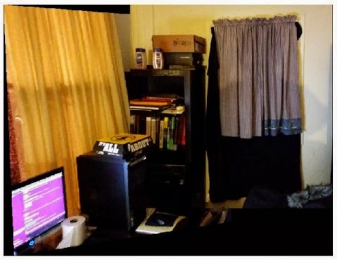
We now apply image stitches to some pictures taken. I took 4 pictures rotating to the right for each image. The first 2 images are shown here with the correspondence points used, the projection, and the merged images in the lower right:

Two images stitched together

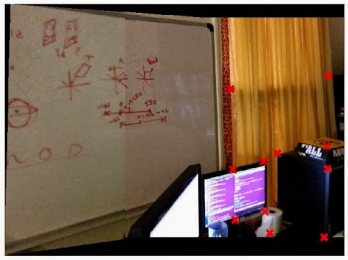
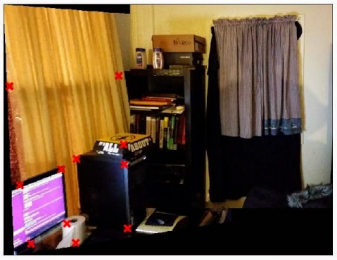
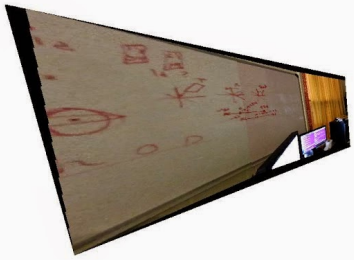
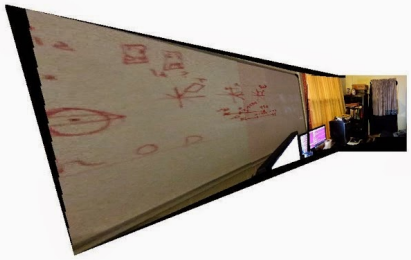
The next 2 images with thier correspondence points, projection, and merged images are shown here:



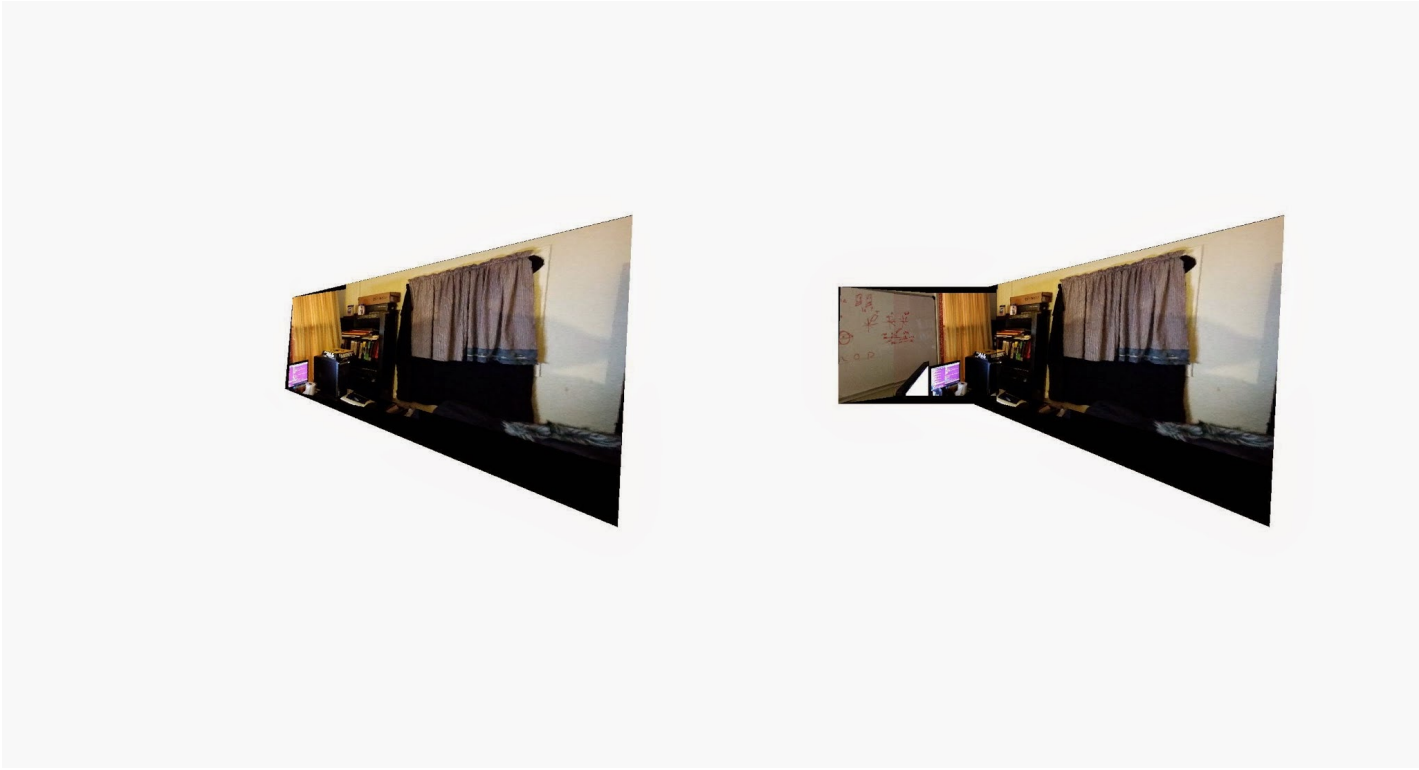
The correspondence points for these 2 stitched images were gathered and all four images were stitched together:

Room images 3 and 4 stitched together

Stitching 2 images together looks good, but stitching all four starts to cause increased distortion (as expected). Projecting the left 2 images to the right 2 images shows a similar distortion, but on the other side.

Projecting the right two stitched images into the left two stitched images.



6. Project onto a rectangle in an image

For projecting onto a surface, we just needed to make sure we draw the smaller projection onto the image. The buildings image is show drawn on the computer screen in the room picture #2:

Building image projected onto the computer screen.

For the coordinates, we just made a square around the projected images border, and another square around the computer screen.