Machine Learning and Computer Vision
Assignment 3

Huihuang Zheng, huihuang@utexas.edu

hz4674 Fall 2015

1 Programming

This section explains usage and implementation of my program.
Usage: run ./src/**main.m**
If you want to test on your own images:
call **twoImageMosaic**(image1, image2, autoMatch, useRansac).

- image1 and image2 are directory and name of two input images.
- autoMatch and useRansac are boolean values to indicate whether use automatically matchin and RANSAC. By default, we use autoMatch but not RANSAC

1.1 Getting correspondences

See **manualCoresp.m**. If you set $autoMatch = false$ in **twoImageMosaic**. My GUI will show the two images. You need to clikc one point in left image, then corresponding one point in right image. Then the next pair of points. Press Enter when you finish.

1.2 Computing the homography parameters

See **homography.m**. The mathematic behind the program:

$$H = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{pmatrix}, p_1 = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, p_2 = \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}$$

We know $wp_2 = Hp_1$, So

$$x' = (ax + by + c)/w$$

$$y' = (dx + ey + f)/w$$

$$w = (gx + hy + 1)$$

From $x' = (ax + by + c)/w$ and $w = (gx + hy + 1)/w$ we have

$$(gx + hy + 1)x' = ax + by + c \Rightarrow x' = ax + by + c - gxx' - hyx'$$

Similarly, from $y' = (dx + ey + f)/w$ and $w = (gx + hy + 1)/w$ we have

$$(gx + hy + 1)y' = dx + ey + f \Rightarrow y' = dx + ey + f - gxy' - hyy$$

We know points x, y, x', y'. We just need to find parameters of H:

$$\begin{pmatrix} x & y & 1 & 0 & 0 & 0 & -xx' & -yx' \\ 0 & 0 & 0 & x & y & 1 & -xy' & -yy' \\ \ldots & & & & & & & \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{pmatrix} = \begin{pmatrix} x' \\ y' \\ \ldots \end{pmatrix}$$

1.3 Warping between image planes

See **warpImage.m**, for every pixel in warped image, I inverse the coordinate to the input image. Then the pixel's value is the interpolate value in input image. In here we need to compute inverse of homography transformation. Say it in mathematic way: $wp_2 = Hp_1$. We know H and p_2 and we want to compute p_1: $H^{-1}wp_2 = p_1$, so $H^{-1}p_2 = \frac{1}{w}p_1$. How can we know w? The third entry of p_1 is one! We just compute $H^{-1}p_2$, then the third entry of $H^{-1}p_2$ is $\frac{1}{w}$, Then we can get p_1.

1.4 Create the output mosaic

See **mosaic.m**, this function is easy, just put one warped image and anther image together.

2 Answer Question

2.1 Output Mosaic

See my result mosaic of two provided UT tower images in figure 1.

2.2 Additional Example

See images of my room in figure 2, and mosaic result in figure 3

2.3 Behavior of Automatically Matching

I found that the most important thing of SIFT automatically matching in this program assignment is the uniqueness of matching. In my experiment, when I started to use *vl_sift*

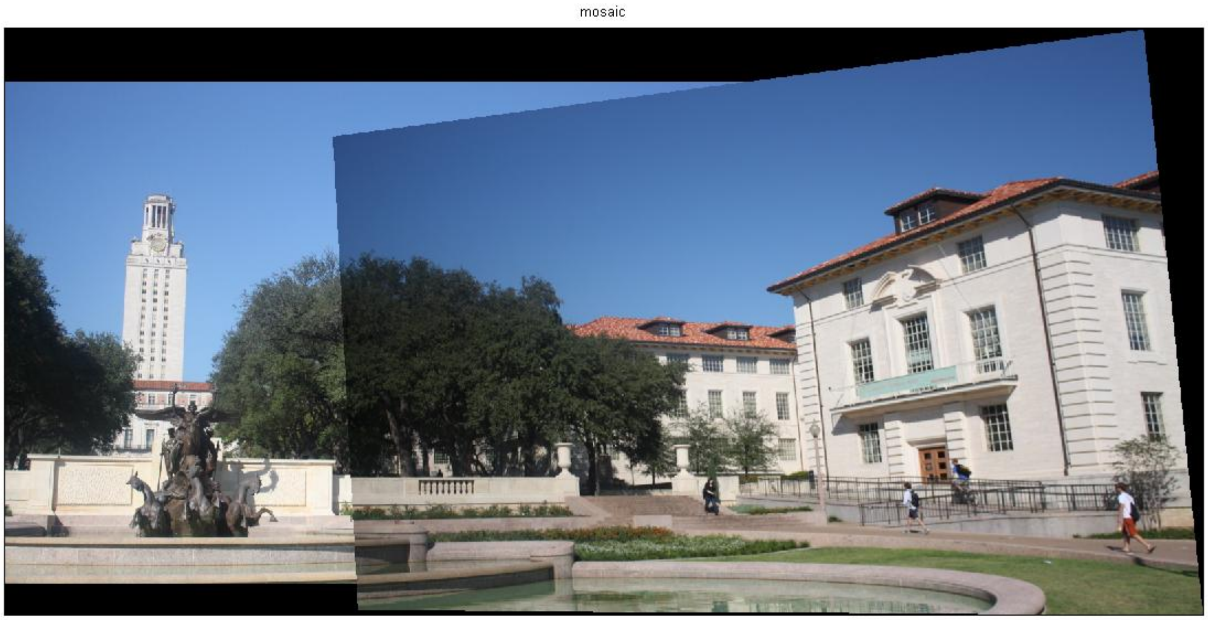


Figure 1: Mosaic of Two Provided Images

and *vl_ubcmatch* at first, because I set small threshold in *vl_ubcmatch* (here, threshold means: A descriptor D1 is matched to a descriptor D2 only if the distance d(D1,D2) multiplied by threshold is not greater than the distance of D1 to all other descriptors). The small value (like the default value of 1.5) will cause result like figure 4. Then I used high value like 5, 10 and got right result in question 2.1 figure 1

2.4 Warp into a frame

I used two images: my photo and a photo of Kristen's class in figure 5. The result is in figure 6. To obtain this is easy. Using manual click, let the points from the one view be the corners of the image you want to insert in the frame, and the let the corresponding points in the second view be the clicked points of the frame. I clicked on four vertexes of my photo and four corner points in Kristen's slide, the output image is just like that.

3 Extra Credit

3.1 RANSAC

See **ransac.m**. In fact, there are many bad matched features of SIFT between uttower1.jpg and uttower2.jpg. For example, I rand chose four matched features to get homography, it

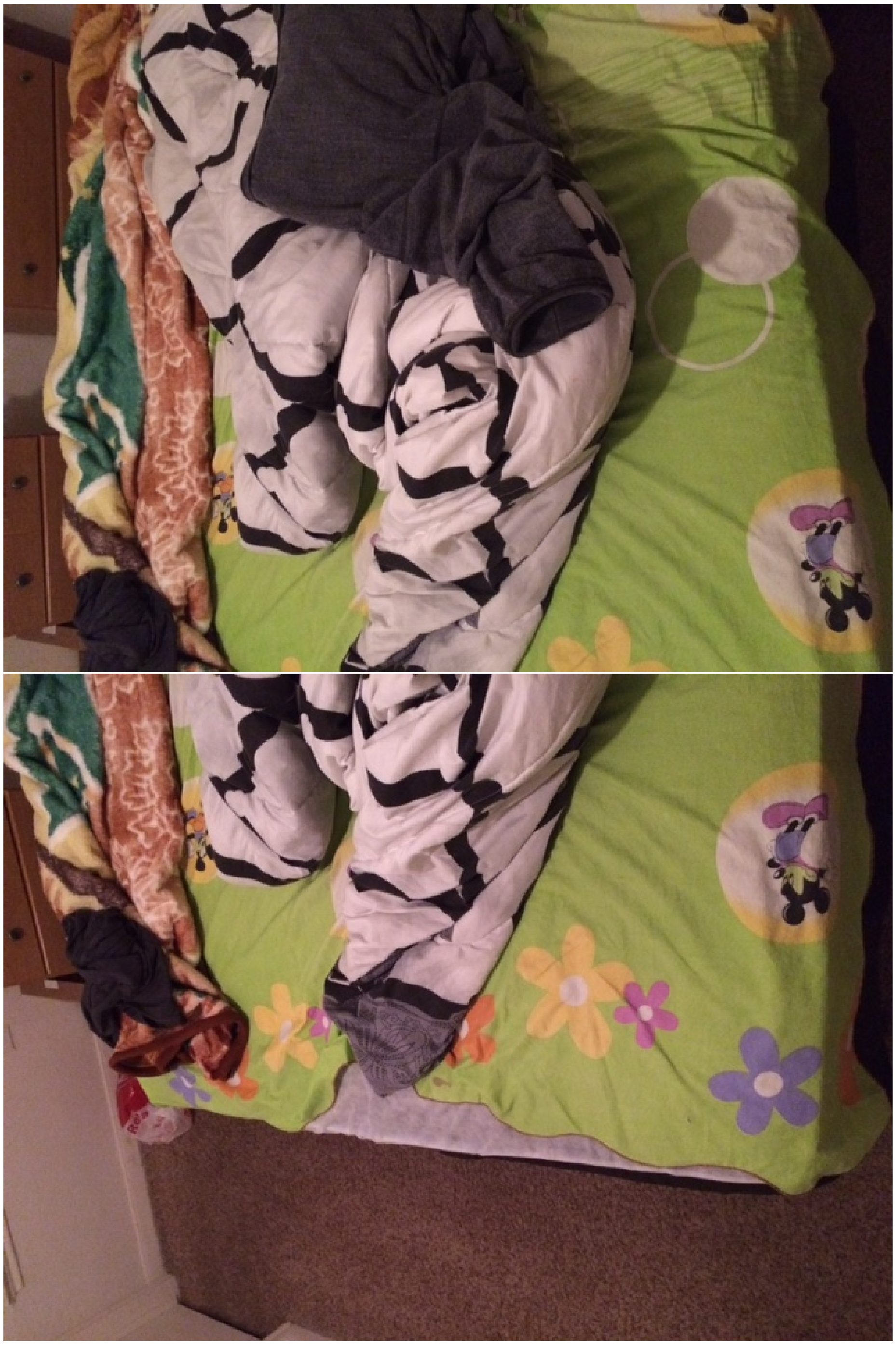


Figure 2: My Room Images

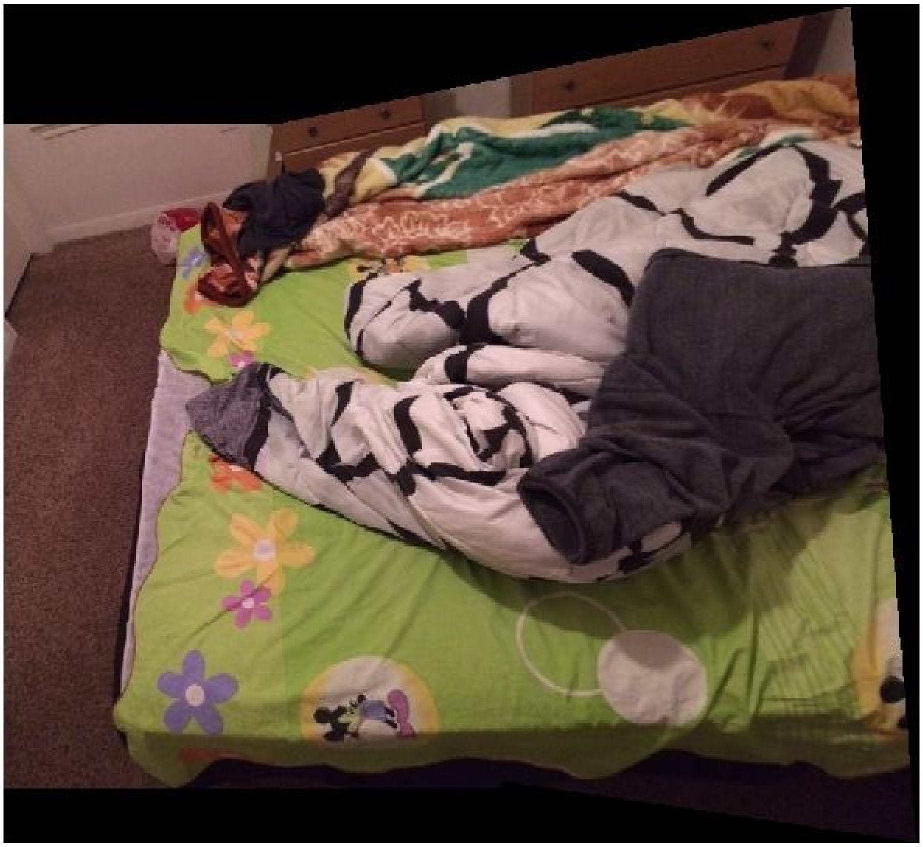


Figure 3: My Room Images Mosaic

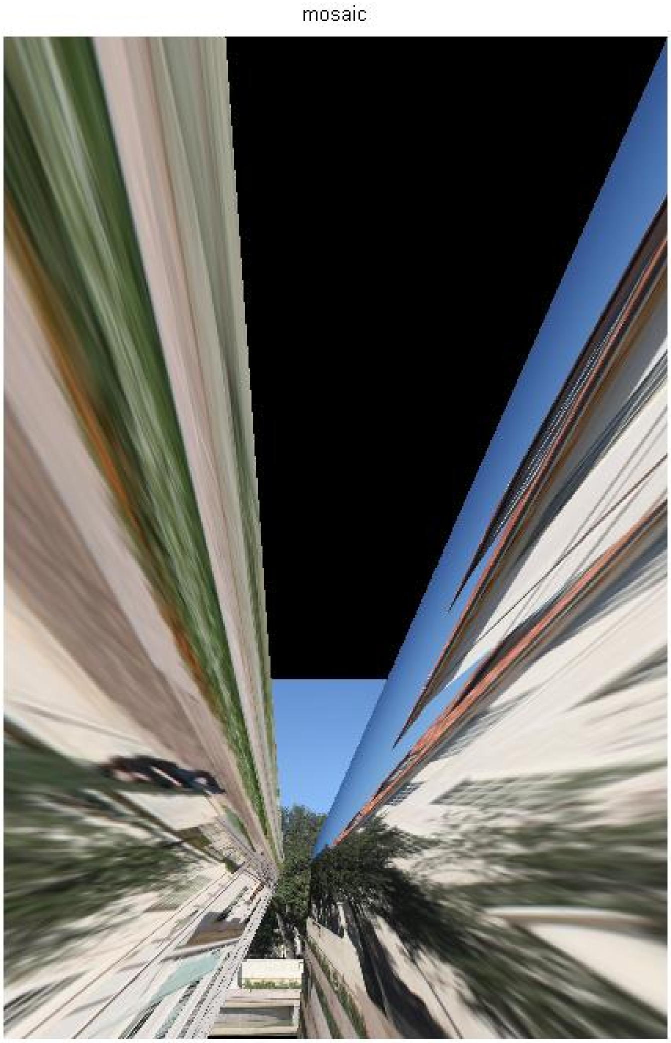
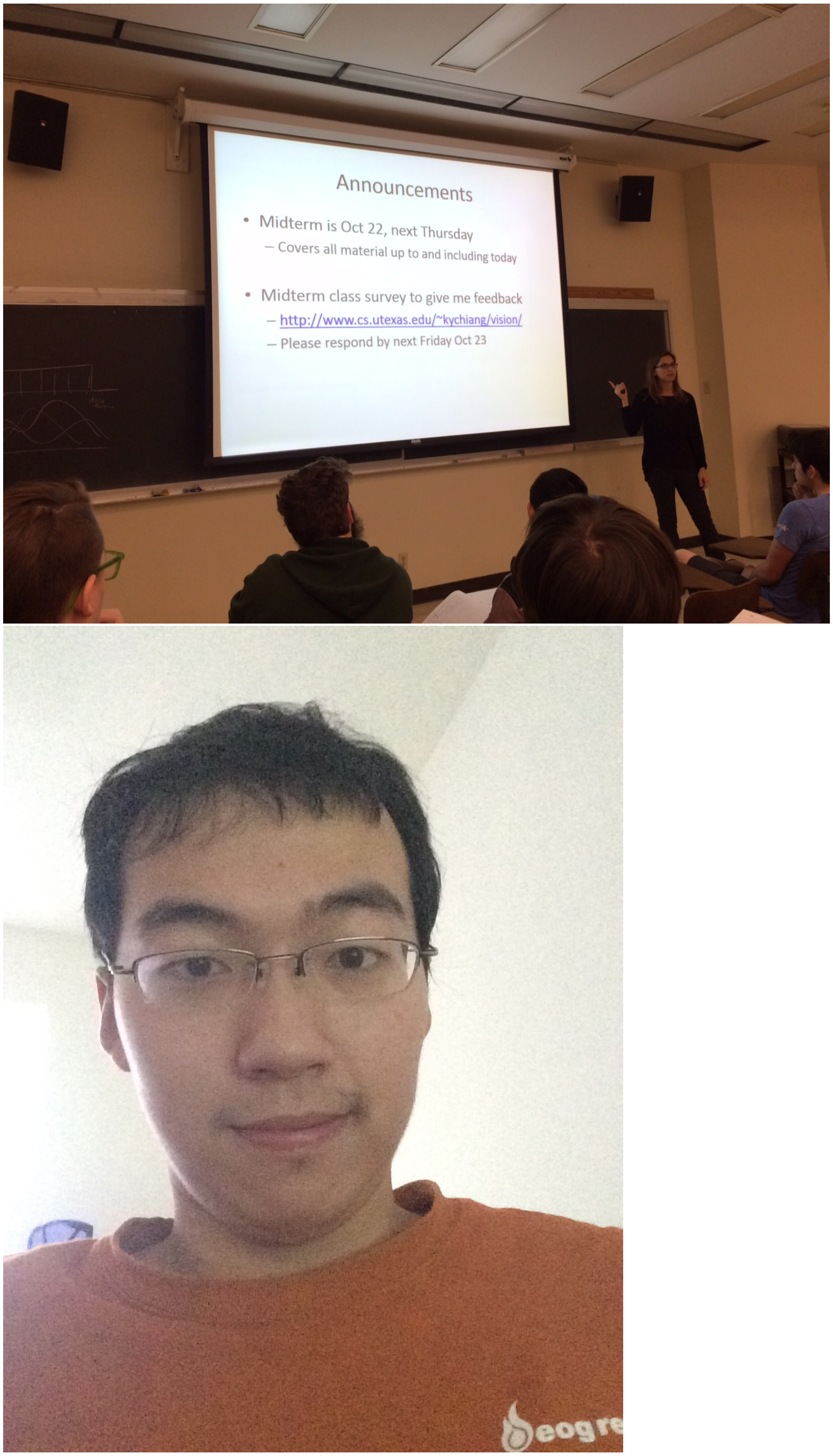


Figure 4: Small Threshold Doesn’t Work



Announcements

- Midterm is Oct 22, next Thursday
  - Covers all material up to and including today
- Midterm class survey to give me feedback
  - http://www.cs.utexas.edu/~kychiang/vision/
  - Please respond by next Friday Oct 23

Figure 5: Two input images

Figure 6: I'm in class image

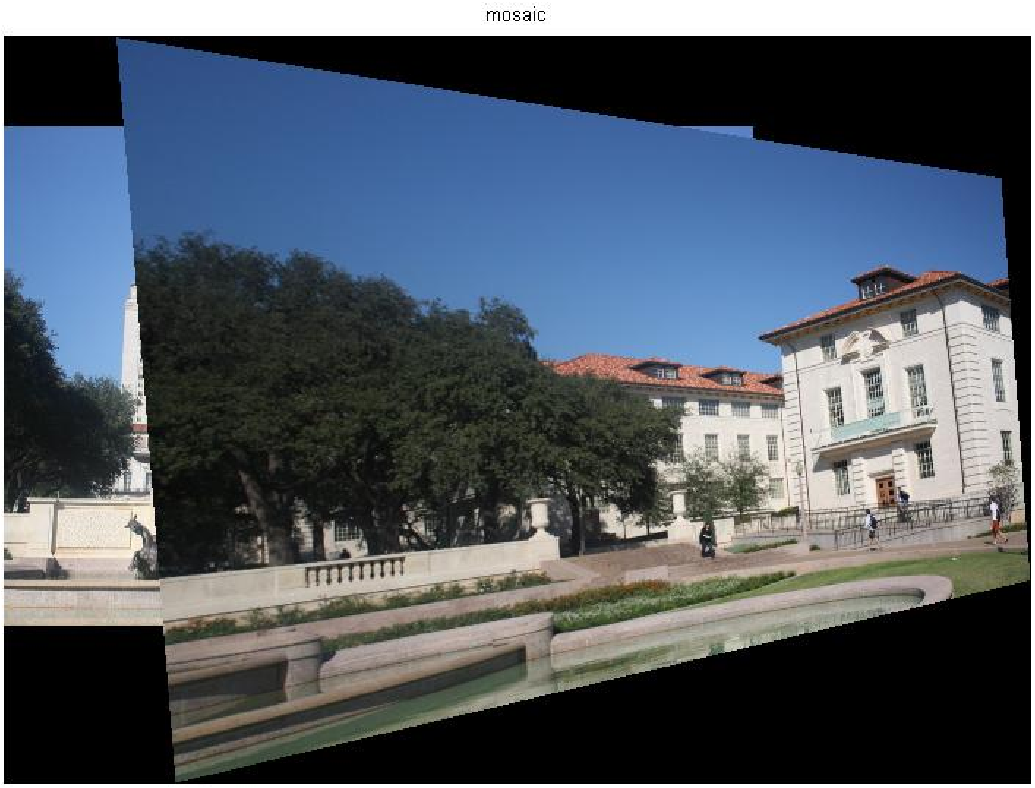


Figure 7: A Bad Example of Feature Matching betweern uttower1,2.jpg

may come out bad example (figure 7). Using RANSAC, I got different result compared to using all matched features (figure 8).

3.2 Rectify

See **rectify.m**. Using my rectify function, you need to choose 4 points in the imags by clicking four corner points, order: top left, bottom left, top right, bottom right. Then press Enter. The output image is square picture.

I used uttower1.jpg (figure 9) as example, the door on second floor of right house is oblique. I choose four points surrounding the second floor, and outputs square result of it (figure 10).

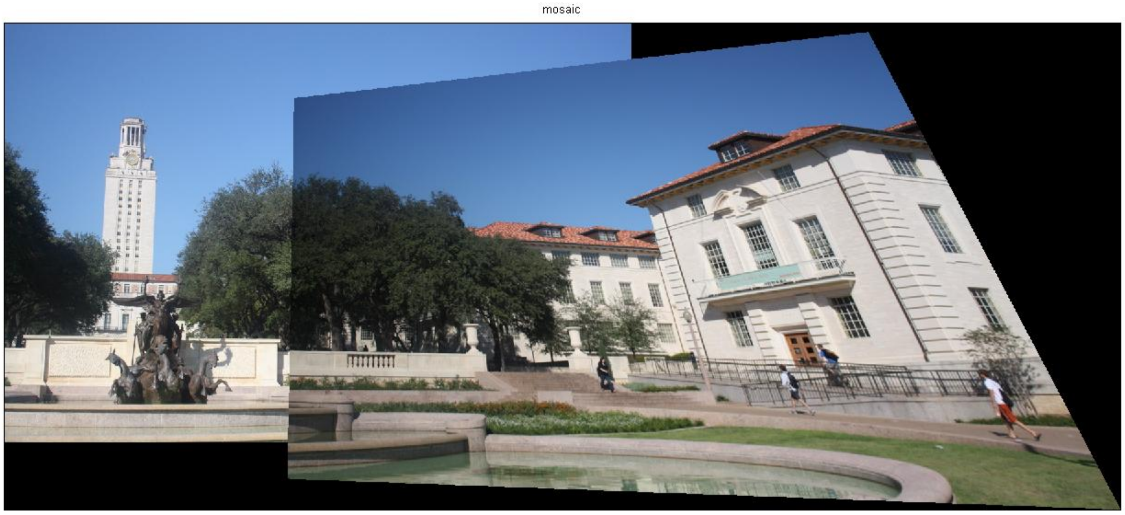


Figure 8: RANSAC result

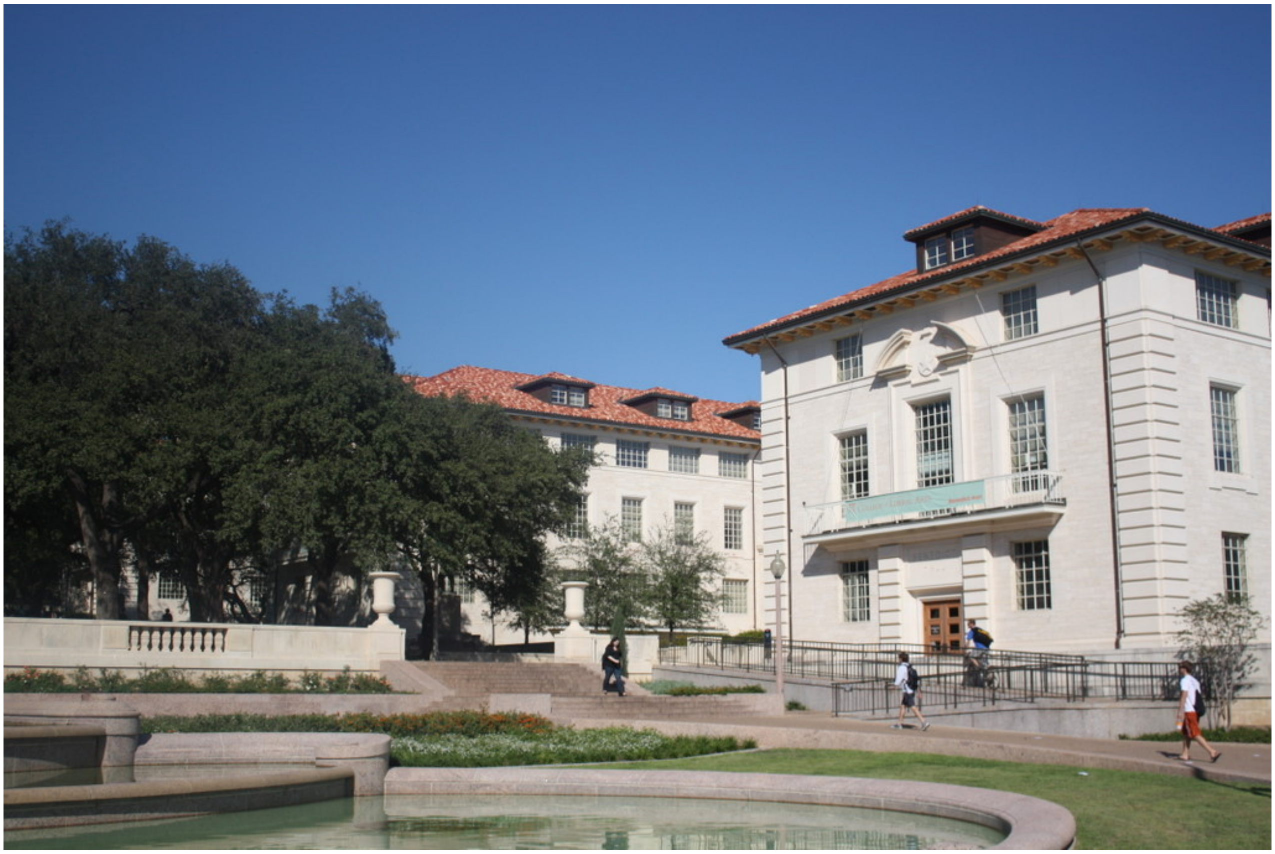


Figure 9: uttower1.jpg

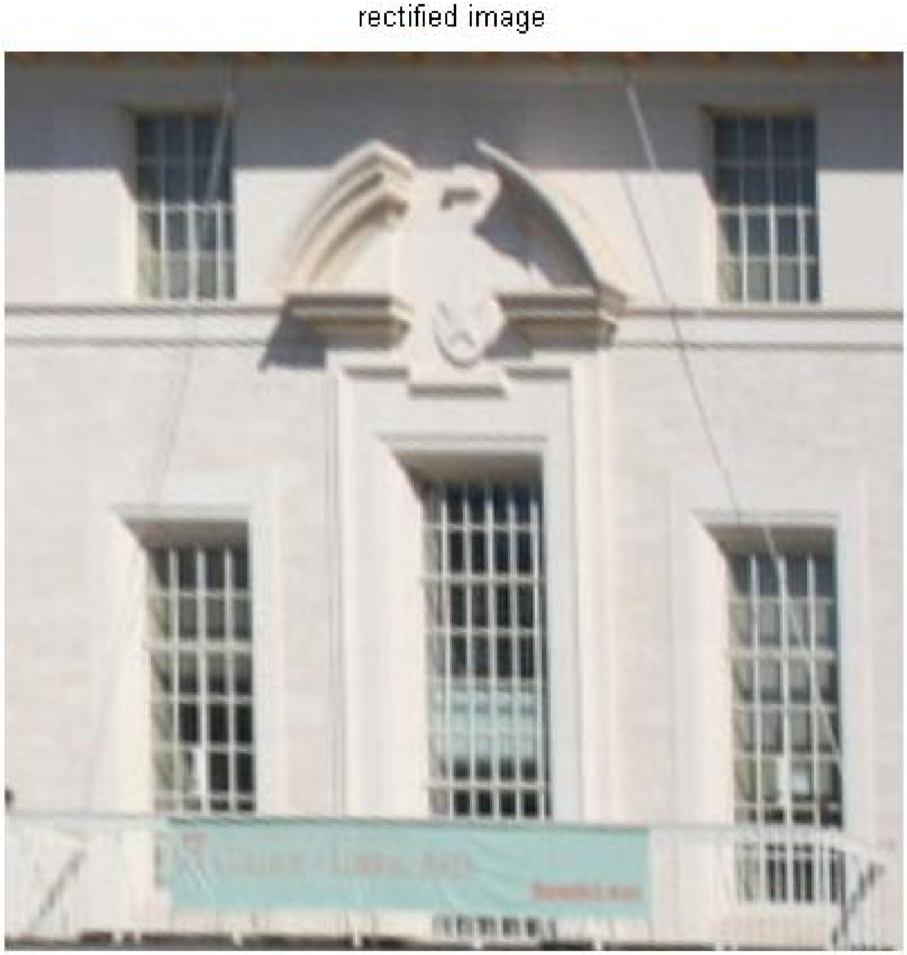


Figure 10: Rectified Image of door in right of uttower1.jpg