Name: Nona Sirakova
CS Login: nonasi
HW 3

Class: CS 376 Computer Vision
Mar 30 2011

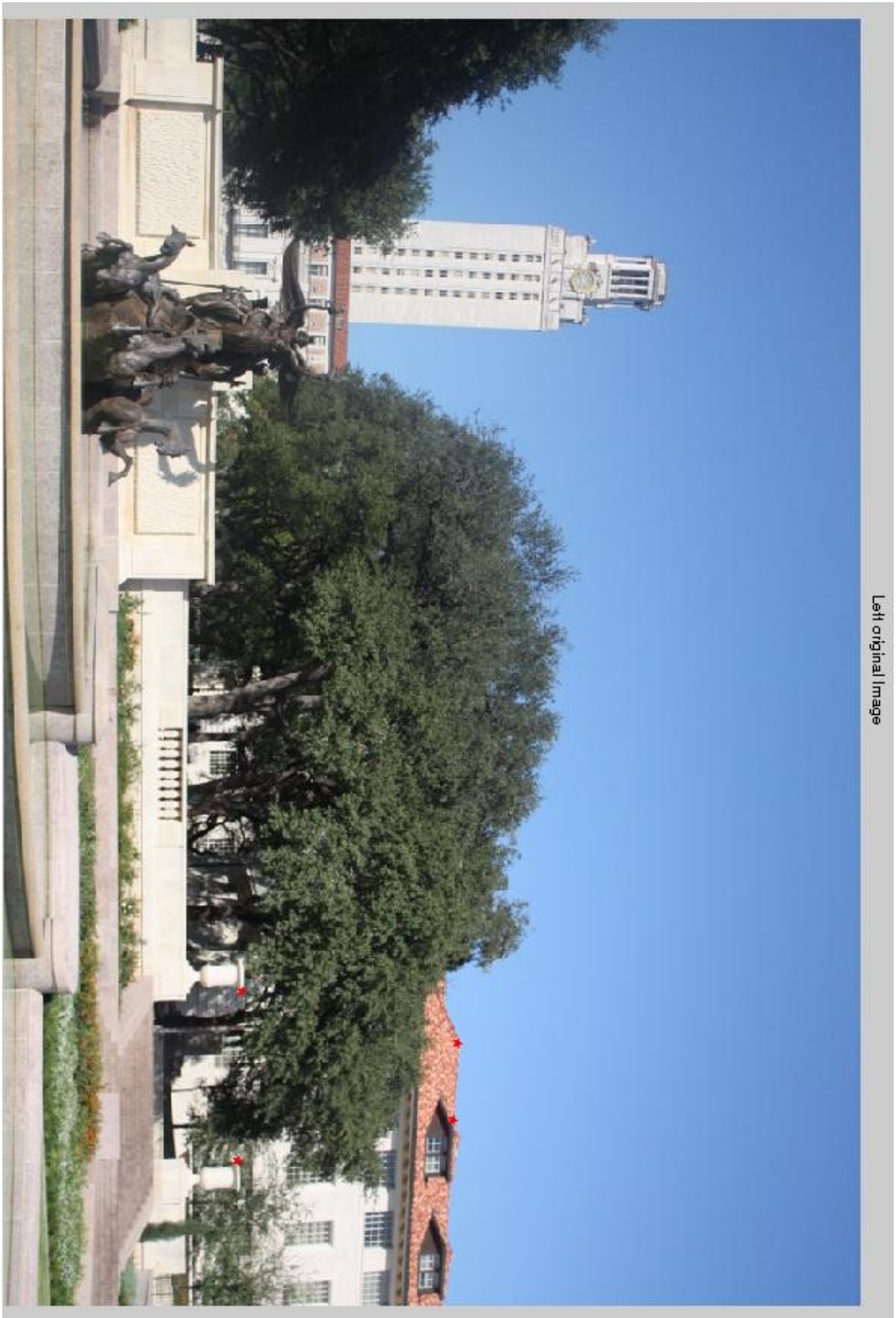
How to run the code:

1. The code is run through the function **ctrl.m**.
2. To pick the number of corresponding points between the 2 images, the user sets this number as the function's ctrl(n, leftiImgage, rightImage)'s parameter n. n = "the number of point pairs we want to pick". If the user picks n<4, n is automatically set to 4. leftImage and rightImage are the titles of the left and right image we will use. I ran the method with the images: ctrl(4, uttower2.jpg, uttower1.jpg).
3. The function **pairs_of_points** is the first function called. It finds pairs of points.
    a. The image "left" appears first. The user can click to pick points on this image. The points will not appear as the user is clicking, but will appear after he is done picking points.
    b. After the user has picked all the points from the left image, red stars will appear everywhere where the user picked a point on the left image. Then to close that image, press enter.
    c. The "right" image will appear next. The user picks points the same way as in the left image. After the stars appear on the right image, the user should press enter to cause the image to disappear and continue with the program.
    d. Note: the k-th point which the user picks in the left image will correspond to the k-th point the user picks in the right image.
    e. The function returns:
        i. x1 – the x coordinates of the points we picked from the 1st image.
        ii. y1 – the y coordinates of the points we picked from the 1st image.
        iii. x2 – the x coordinates of the points we picked from the 2nd image.
        iv. y2 – the y coordinates of the points we picked from the 2nd image.
4. Ctrl then calls the function **find_homography_matrix** (x1, y1, x2, y2, left, right, n). Where x1, y1, x2, y2 have the same meaning as they did in 3; n is the number of pairs we have; "left" is the first image and "right" is the second image. For this function we use the Ax =b equation as it is explained in the handout for this assignment. And the matrices we used are named the same way. The function outputs H – the homography matrix.
5. The next function which gets called is the function warp (h, warp_from, warp_to) where H is the homography matrix, warp_from is the image we will apply H to and warp_to is the image to whose plane all pixels of warp_from will get mapped using the transformation matrix.

Results and explanations:

Experiment 1 (good matching):

Left image:



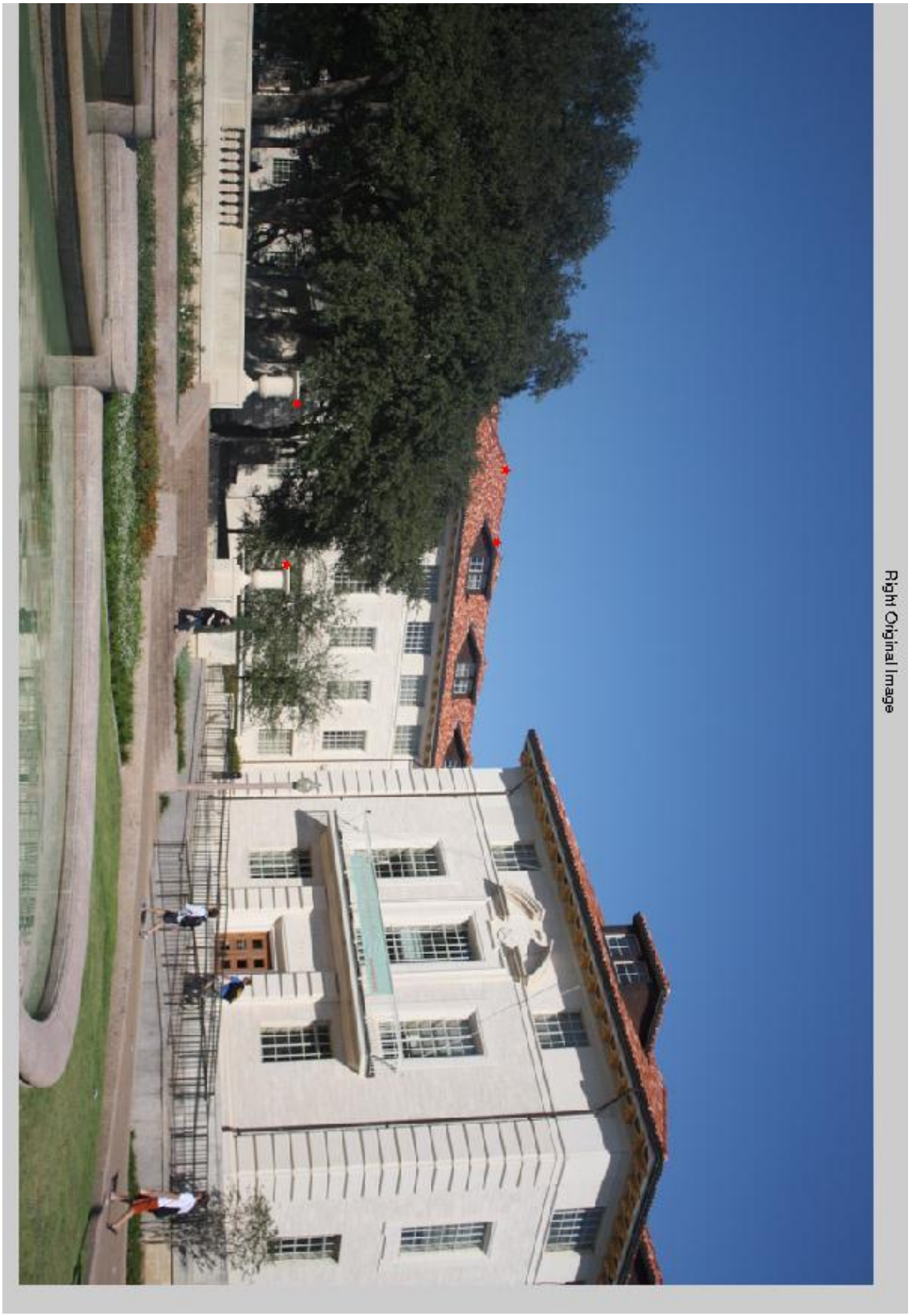
The stars in red show the locations of the points I picked.

```
x1 =
[775.1923,
815.7220,
877.2670,
910.2912]';

y1 =
[493.6110,
322.4857,
325.4879,
498.1143]';
```

Right image:

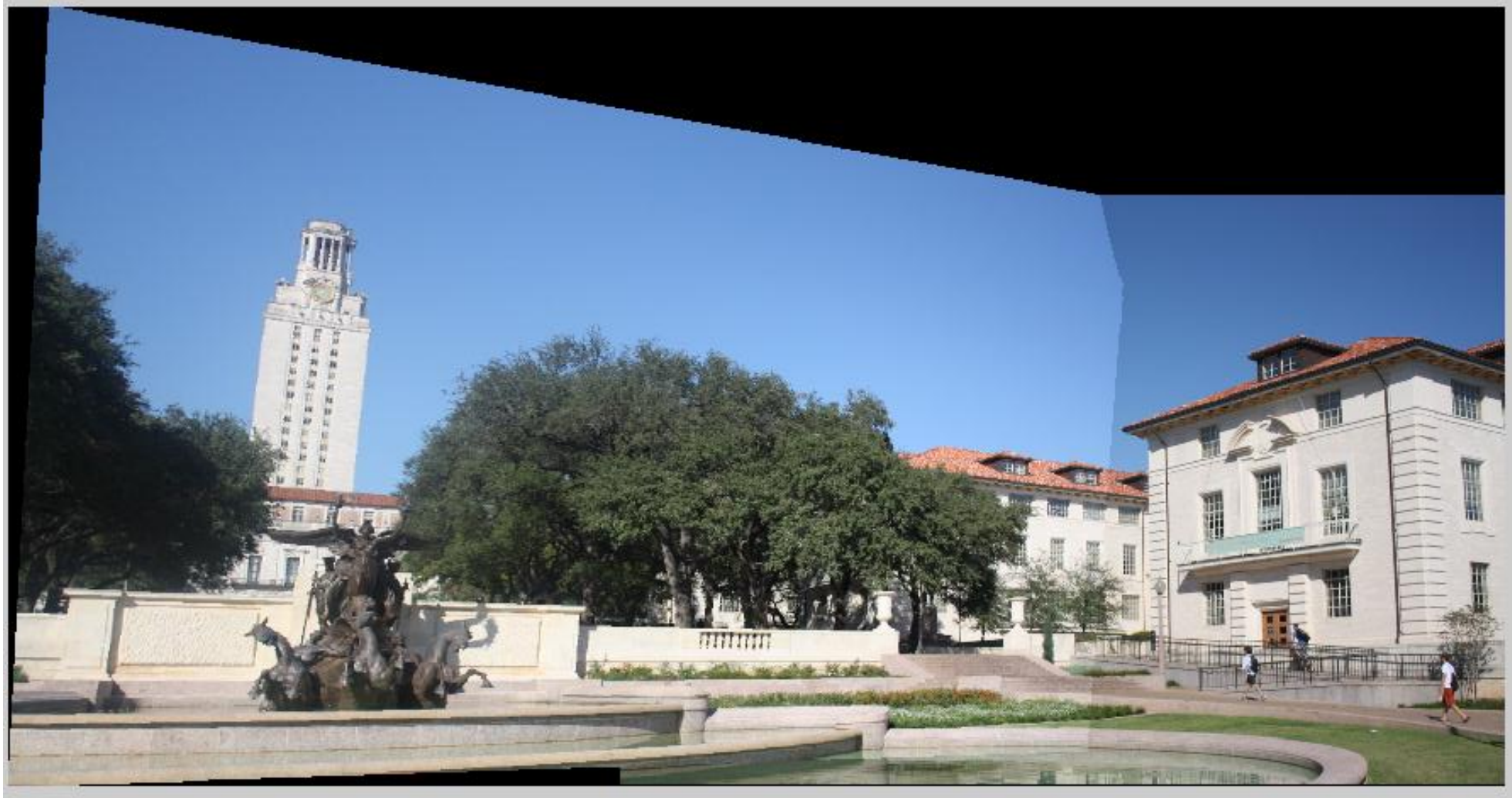
The stars in red show the locations of the points I picked.

```
x2 =[321.8604,
374.3989,
432.9418,
450.9549]';

y2 = [462.0879,
293.9648,
301.4703,
469.5934]';
```



Warped Image:
Notice, in this case we do not have the black speckles in the top left corner (I handled this case by using the following approach which we studied in class: Apply H to each pixel, p, in the image. Obtain some new locatioln (x_new, y_new) where this pixel should fall on the plane of the other image. If the resulting pixel's x_new or y_new is not an integer, then for the coordinate which is not an integer – apply the color of p to the two immediate neighbors of the (x_new, y_new). So if x_new is not an integer – we will apply the color to (floor(x_new), y_new) and (ceiling(x_new), y_new). If y_new is not an integer we will do the same thing but apply the floor and ceiling to y_new. If both are not integers – we will apply floor and ceiling to both and get 4 combinations out of it. This distribution of colors handles the holes in this case, but not in the first experiment because in the first experiment the corner which is very stretched needs for the points in that location to cover more then just their immediate neighbors.

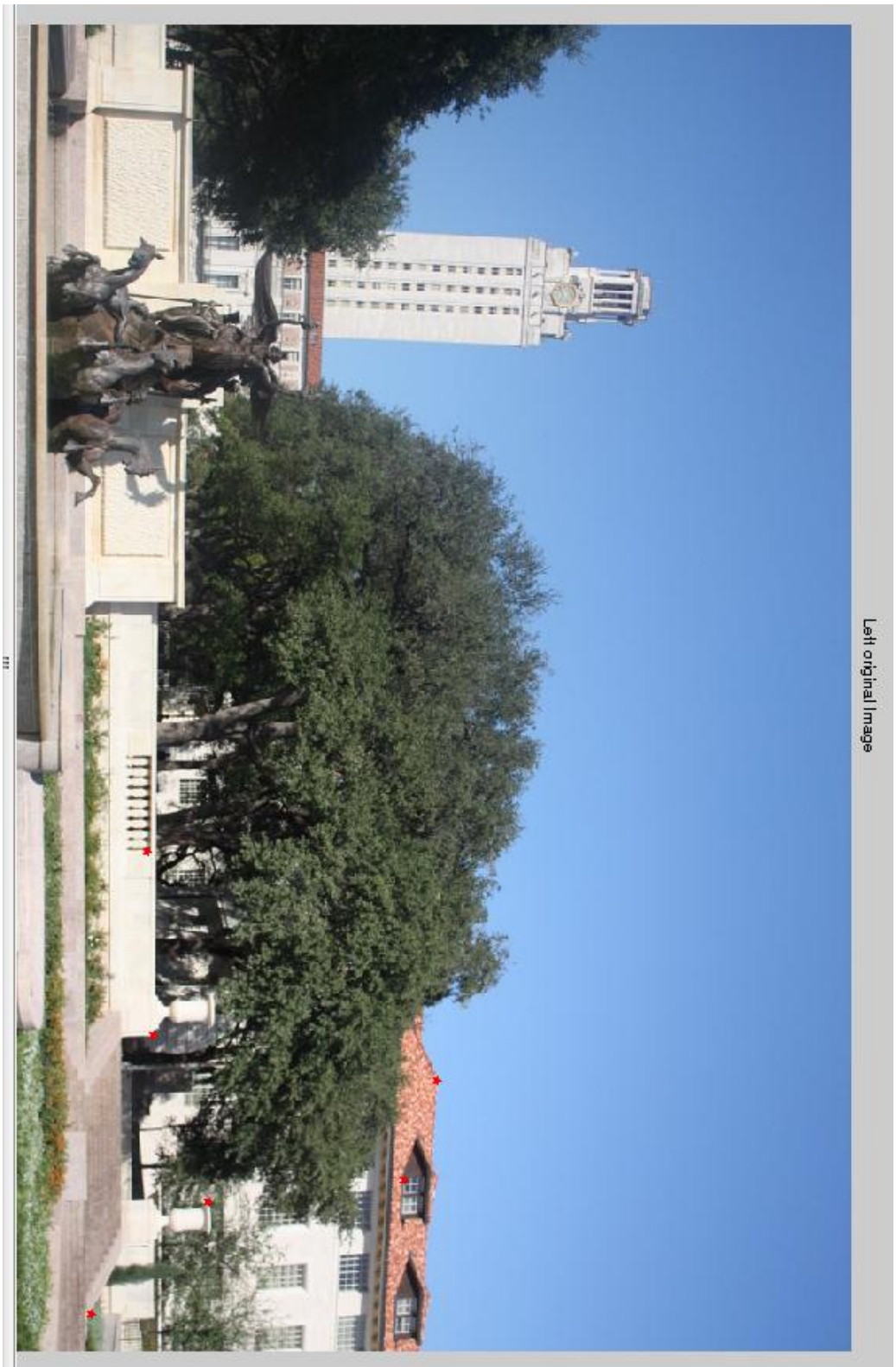
Blended Result (Warped Image)



<u>Expriment 2(more then 4 points):</u> The point of tis experiment is not so much to achieve great matching, but to show that the code works with n>4. In particular, the number of pairs we picked for this experiment is 8.

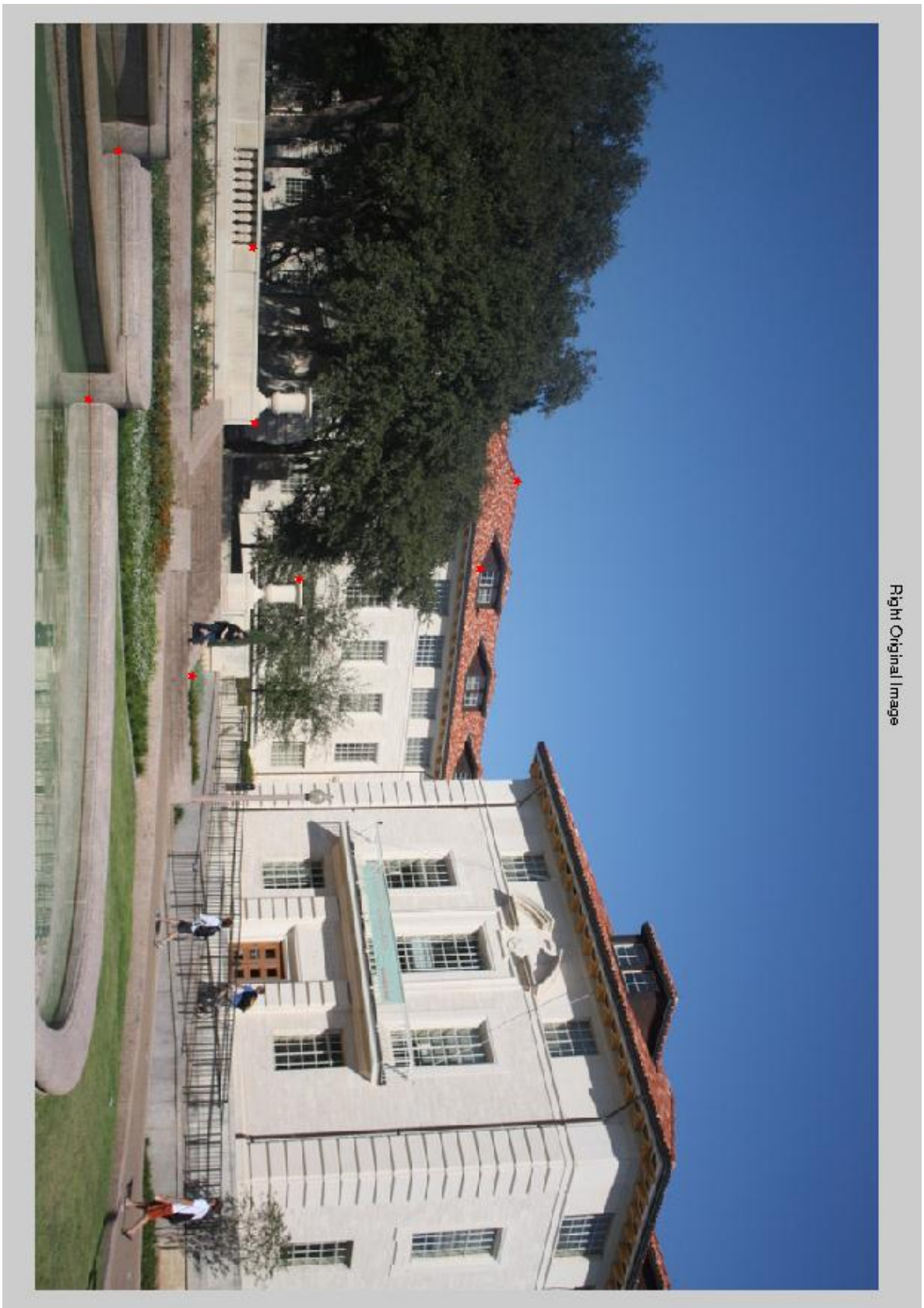
Left image:
The stars in red show the locations of the points I picked.

```
x1 =[773.9085,
574.3967,
996.2559,
639.2981,
781.1197,
909.7207,
892.8944,
815.9742]';

y1 =[675.5211,
650.2817,
587.7840,
544.5164,
539.7089,
497.6432,
346.2066,
320.9671]';
```

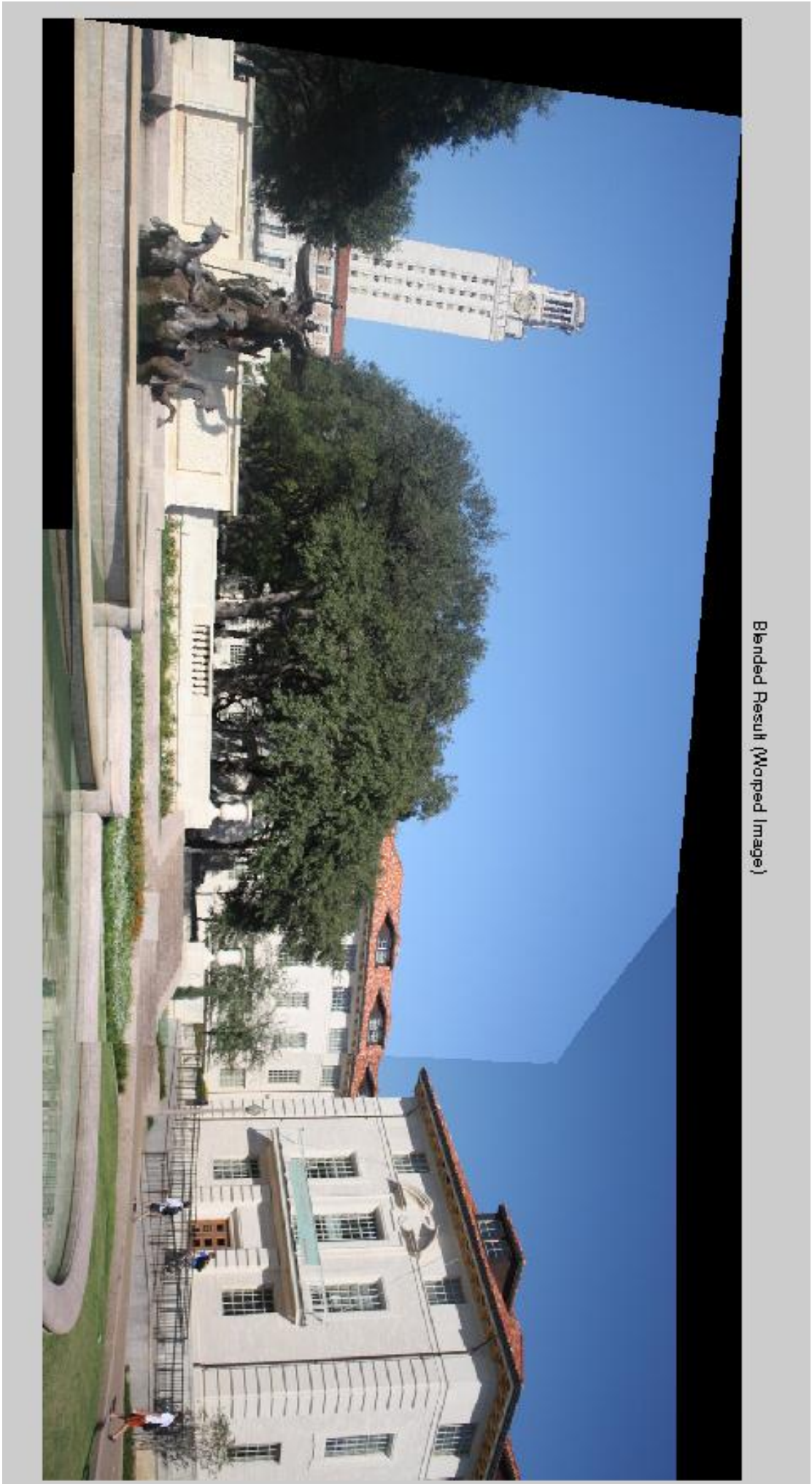


Right image:
The stars in red show the locations of the points I picked.

```
x2 =[305.3484,
104.2011,
528.7254,
182.5845,
324.4061,
450.6033,
442.1901,
371.2793]';

y2 =
[640.7187,
616.7011,
556.5352,
507.2582,
506.0563,470.0
000, 323.3709,
293.3239]';
```



Warped Image:



Experiement 3 – My own image:

To run this experiment I used the images houseLeft.JPG and houseRight.JPG.

`Left image:`
The stars in red show the locations of the points I picked.

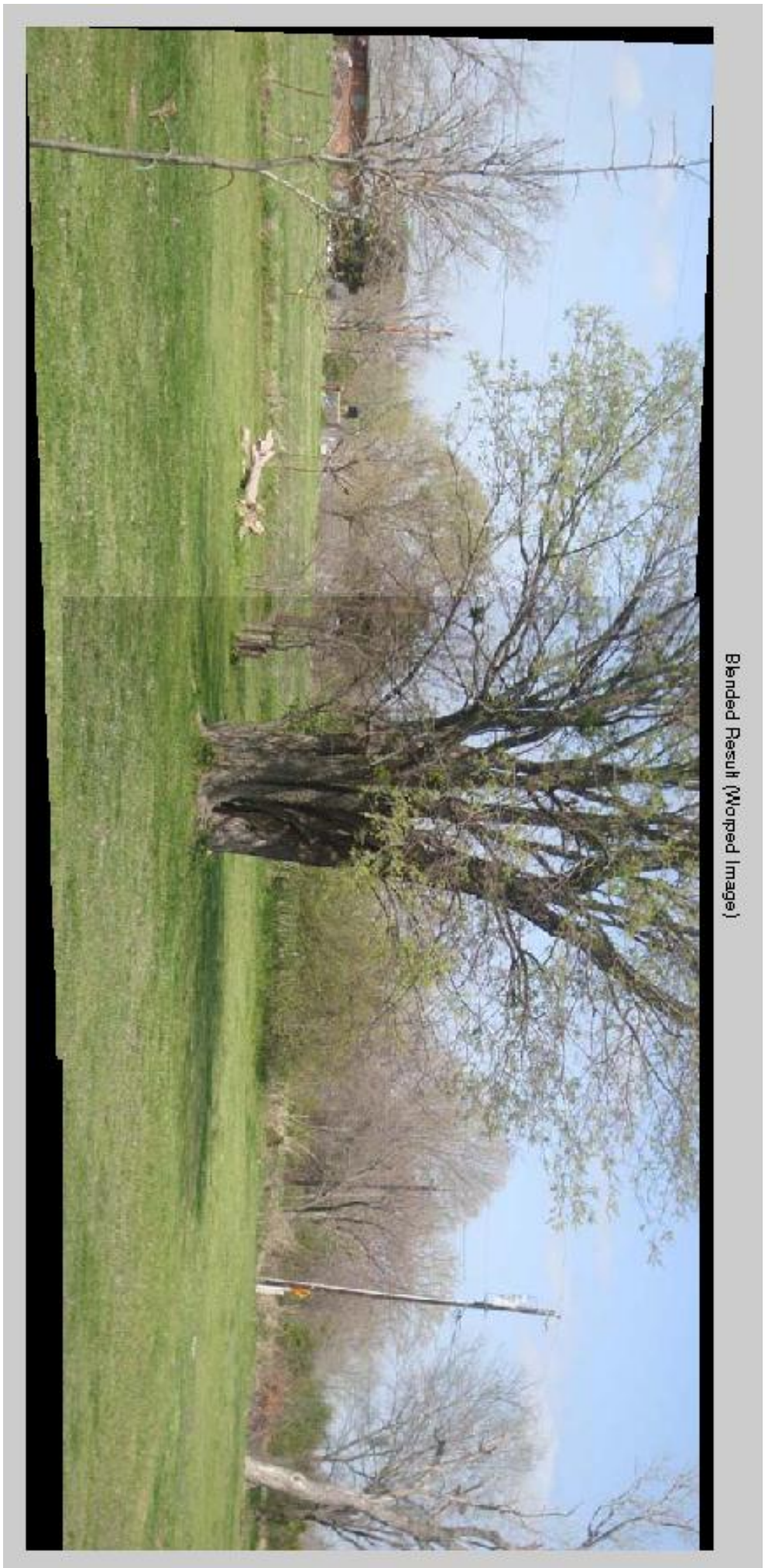
```
x1 =[ 416.6583,
534.8423,
603.5733,
484.5512]';

y1 =[322.6420,
361.1985,
108.9051,
177.6362]';
```

Right Image:

The stars in red show the locations of the points I picked.

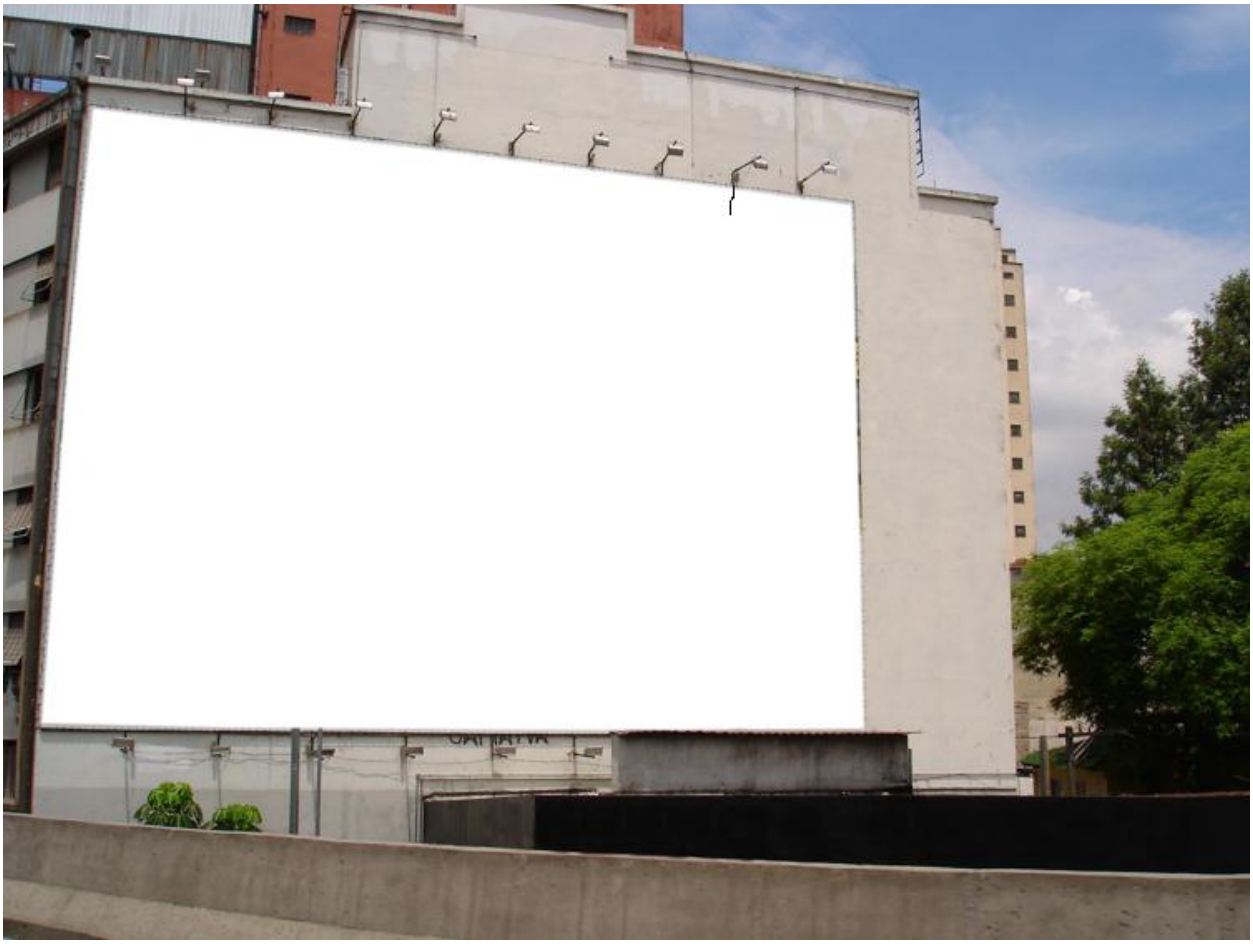
```
x2
=[40.3137,
162.6886,
233.0960,
112.3976]';

y2
=[330.1857,
367.9040,
117.2870,
183.5035]';
```

Warped Image:



Experiment 4: Warp one image into a “frame” region in the second image.

Materials used for this experiment:
Billboard picture taken from: http://www.freakingnews.com/Billboard-Pictures--2340.asp
Billboard.jpg
Original Billboard Image:

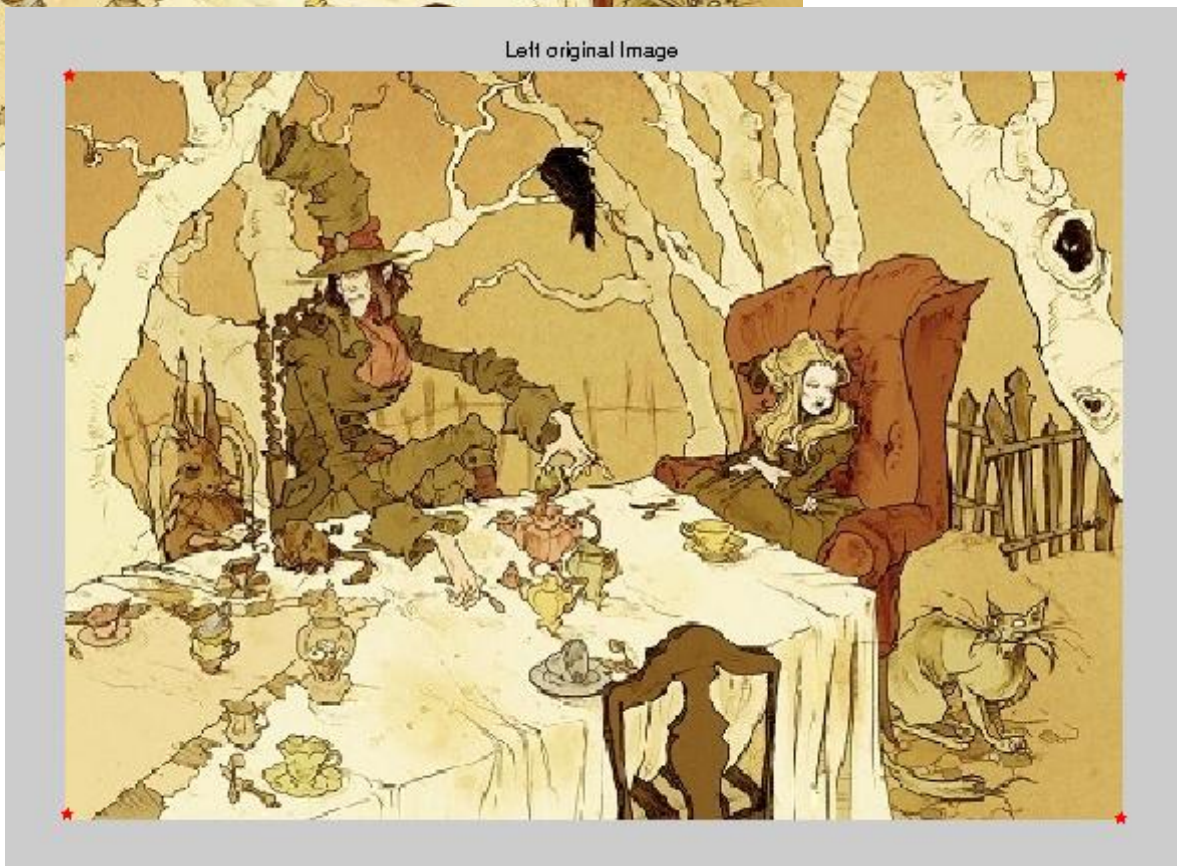


Alice in Wonderland picture taken from:

http://www.google.com/imgres?imgurl=http://fc07.deviantart.net/fs70/i/2010/128/a/8/A_Mad_Tea_Party_by_MirrorCradle.jpg&imgrefurl=http://sudasuta.com/%3Fp%3D2385&usg=__bsoMSH2oqsPN6g5PT155cPRR1M8=&h=724&w=1024&sz=307&hl=en&start=0&zoom=1&tbnid=5aM2VPRHBrAgAM:&tbnh=139&tbnw=190&ei=lruLTa6EJYy70QGuqaCwCw&prev=/images%3Fq%3Dmadd%2Bteaparty%2Bmirror%2Bcradle%26um%3D1%26hl%3Den%26biw%3D988%26bih%3D646%26tbs%3Disch:1&um=1&itbs=1&iact=hc&vpx=127&vpy=134&dur=94&hovh=189&hovw=267&tx=171&ty=56&oei=lruLTa6EJYy70QGuqaCwCw&page=1&ndsp=16&ved=1t:429,r:0,s:0

Or by typing “madd teaparty mirror cradle” in the google search browser (since the url is increadibly long).
Alice.jpg

Original image:

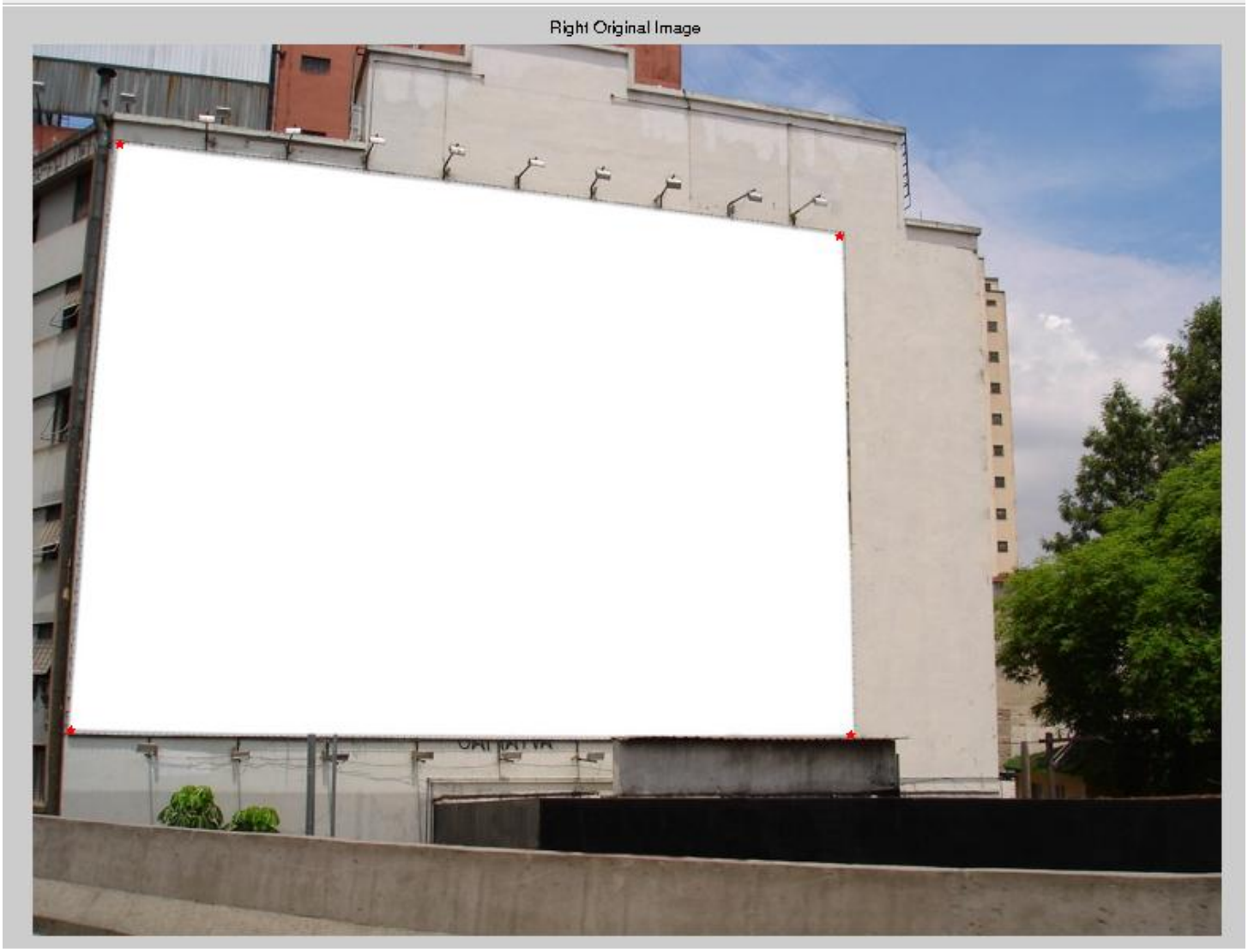


The experiment itself:
Left image:

The stars in red show the locations of

the points I picked.

Left image:
The stars in red show the locations of the points I picked.



Warped Image:

Extra Credit:

Rectify tile image:

The code:

This code runs through the method warp_floor(image_name); where the image name is the name of the image we will use. The image I used for this experiment is 'floor.jpg' and floor2.jpg. The images are shown below.

The method works by having the corners of one of one square tiles be selected by the user and then mapped to a top down view of the floor. (At first I made it hardcoded so that the user had no input, but that takes much of the fun out of it and it is harder to run the code on different floor images) . The selecting of the points happens when the warp_floor function calls its helper function single_pts which does the same thing as pairs_of_points.m but instead of calling 2 images for the user to pick points from, it only calls one.

Then, warp_floor calls find_homography_matrix, which here works the same way as in the rest of the assignment. And finally the function calls warp_single which is a helper function which does the same thing as warp.m but for a single image. Finally we crop the resulting image so it contains the smallest rectangle in which our floor fits and we display the floor.

How do we decide what the top down view is? We use the knowledge that we selected a square of the tiles so the points we have get mapped to are a predetermined square.

Below are the original images I used and the sites I got them from:

Floor.jpg: http://www.google.com/imgres?imgurl=http://www.aaffordableservices.com/wp-content/uploads/2009/08/tile-floor-1.jpg&imgrefurl=http://www.aaffordableservices.com/%3Fpage_id%3D688&usg=__spXfcutiGcvIxBH_GA_33t2q528=&h=480&w=640&sz=67&hl=en&start=38&zoom=1&tbnid=3ZPEYu_Y0aEmiM:&tbnh=117&tbnw=156&ei=N9qTTa7TCYW10QGrmqHwCw&prev=/images%3Fq%3Dfloor%2Btile%26um%3D1%26hl%3Den%26sa%3DN%26biw%3D1041%26bih%3D720%26addh%3D36%26tbs%3Disch:10%2C1000&um=1&itbs=1&iact=hc&vpx=117&vpy=296&dur=50&hovh=194&hovw=259&tx=106&ty=84&oei=UtmTTaO9BcibtwfvmOTnCw&page=3&ndsp=22&ved=1t:429,r:0,s:38&biw=1041&bih=720

Tile image itself:

Cropped image which I use:

Floor.jpg after I cropped it so it can be applied to the experiment:

Experiment Results:

The input image of the tiles: (note the red stars represent the locations which I used to calculate the homography matrix)

Results:

Discussion of results:

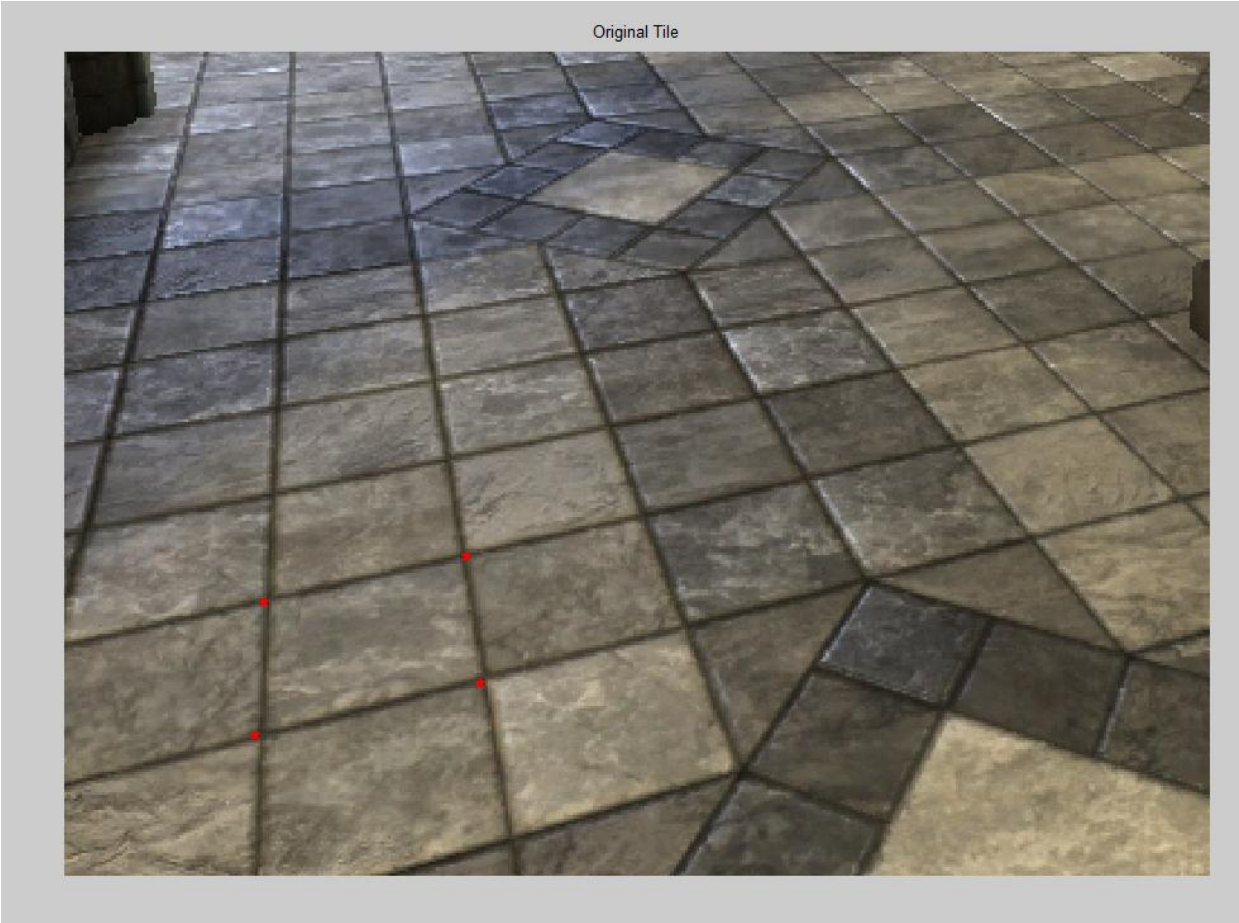
The warped image is a top down view of the floor. The view does not have a nice rectangular shape and instead has black regions on the sides but this is because those areas have nothing to map to them. Otherwise the floor got reconstructed correctly.

Floor2.jpg:
http://www.google.com/imgres?imgurl=http://www.dregsld.com/tutorials/FloorTile/NewFloorTile1.jpg&imgrefurl=http://www.dregsld.com/tutorials/FloorTile/tiletut01.html&usg=__A8gAqSV9WSkk6lieY2Esd4-bRB4=&h=541&w=600&sz=337&hl=en&start=0&zoom=1&tbnid=IzxpKJVwwq4cyM:&tbnh=130&tbnw=128&ei=kvaTTYHHL5K3tweNqbztCw&prev=/images%3Fq%3Dfloor%2Btile%26um%3D1%26hl%3Den%26sa%3DX%26biw%3D592%26bih%3D328%26tbs%3Disch:1,isz:m0%2C114&um=1&itbs=1&iact=rc&dur=623&oei=k_aTTaXTE5Gztwewt6CKDA&page=1&ndsp=17&ved=1t:429,r:2,s:0&tx=43&ty=90&biw=1041&bih=720

The image itself:

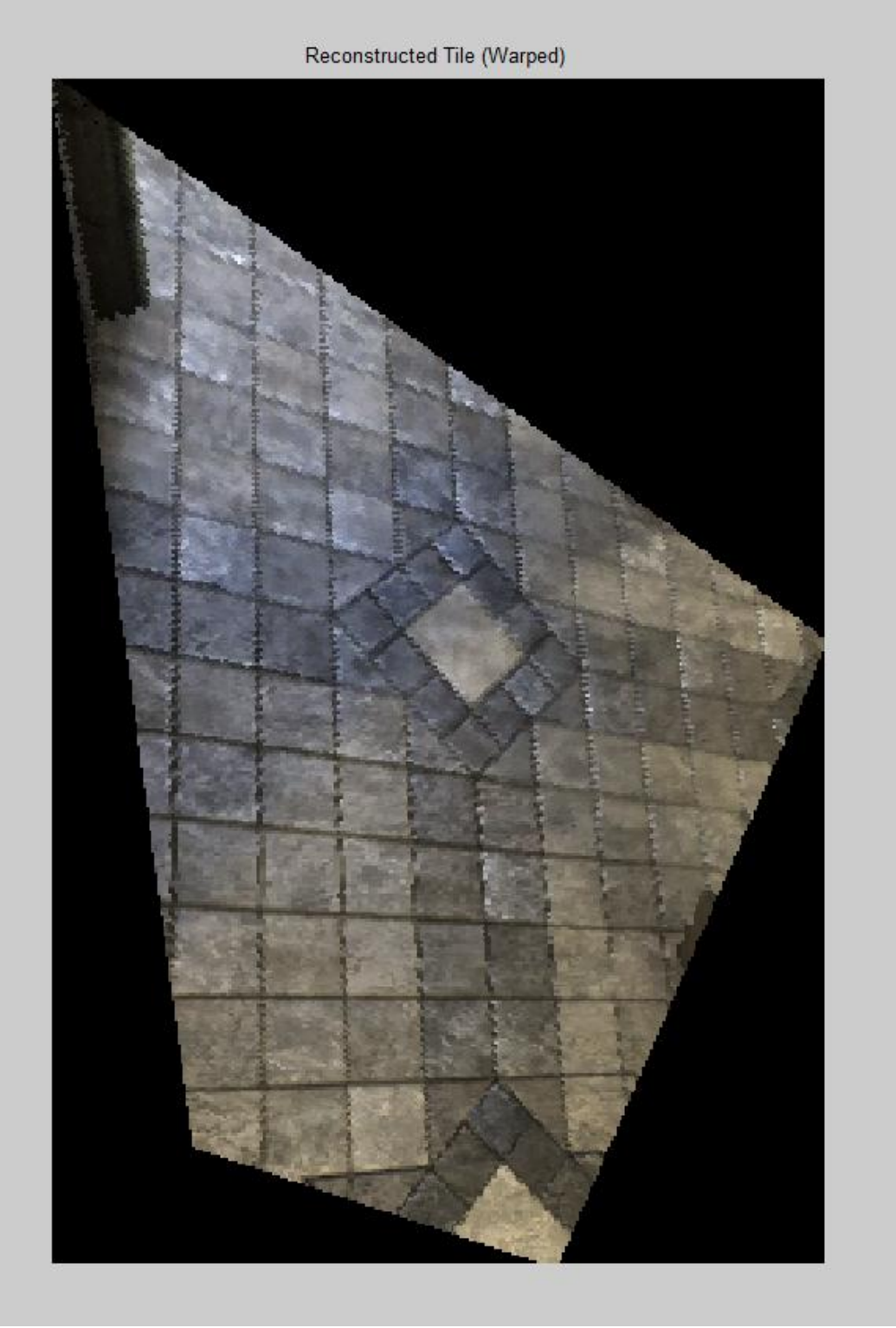
The image after I cropped it for use in the experiment:

Results of the experiment:



The red stars show the tile points which the user picked.

Warped result:



Discussion of result:

Notice that the view of the image at the bottom left looks exactly as though one is looking top-down on the image. But as we move to the right the image seems to slant. This is because the picture shows a projection in which the tiles closes to us (the ones which end up on the bottom left corner) seem like they are slanted a little to the left, whereas the tiles in the top right corner seem to be slanting to the right. So it seems like the floor is sort of a hill with the highest part being the darker gray tiles. This image supports this view.