NAME: PRAJWALA TM
ROLL NO: 14CO133

ASSIGNMENT-2 REPORT

The input images originally have been clicked with a camera of 16MP resolution and have been rescaled from (1280,960) to (800,600) keeping the height-width ratio constant. The rescaling of the images induces noise.

The two images clicked have around 40% of overlapping region as specified in the question.

STEP-I : NOISE REMOVAL

The images *'b1.jpg'* and *'b2.jpg'* are input in both grayscale and RGB colorspaces. Grayscale images are used at this stage for corner point detection.
Due to the rescaling of the images, noise is induced which can be best removed with a Gaussian filter. It performs best in comparison to median or any other noise remval filter for this purpose. Different kernels have been experimented with, and 3x3 filter seems to work best for the images under consideration.

STEP-II : CORNER POINT DETECTION

The denoised grayscale images are input to the Harris Corner Detector. Different sets of parameters have been experimented with, here again, and optimal values of 0.005 and 0.04 for threshold and the harris free parameter have been chosen. Corner points in the two images have been obtained and they have been written to files *'corners1.txt'* and *'corners2.txt'* respectively. They have also been plotted in the input images as *'cimg1.png'* and *'cimg2.png'* respectively. *'plotted1.png'* and *'plotted2.png'* consist of overlapping points in the images. The plotting has been done such that a few of the neighbouring pixels have been colored around the precise corner pixel for visibility.

In order to obtain the overlapping points for the purpose of application of projective transformation, the co-ordinates of **four** points in the overlapping region have been manually chosen. 3 out of the 4 points chosen need to non-collinear for solving the linear equation of the projective transformative matrix. The points have been obtained carefully using the text files and *GIMP image editor of Ubuntu OS*.

STEP-III: PERSPECTIVE TRANSFORMATION MATRIX COMPUTATION

Note: All co-ordinates have been transformed to homogenous co-ordinates in order to apply the DLT algorithm
The projective transformation model of the pinhole camera is applied here. Since the transformation matrix mapping from world to image 1 plane and from world to image 2 plane is linear, the mapping from image 1 plane to image 2 plane is also linear. This is applied here to obtain the perspective transformation matrix (homography matrix) from image1 to image2. Its a generic model which considers rotation, scaling, translation and preserves straight lines as straight lines. The DLT algorithm uses homogeneous coordinates where the homography is a linear transformation and finds a closed form solution for the algebraic error.

Numpy library of python is used for computing the transformation matrix using SVD.

```
U,S,V = np.linalg.svd(A)
```

The transformation matrix thus obtained is used for obtaining *'perspective.png'*. Each of the co-

ordinates are multiplied with the matrix to transform it to the plane of the second image.
The corresponding pixel intensities are filled to obtain the warped image.

Note: *Direct multiplication of the homography matrix results in co-ordinates which are negative and will be out of bounds. For complete display of the warped image, its boundaries are calculated initially ie: (0,0), (h-1,0) ,(h-1,w-1) and (0,w-1) are projected and the size of the warped image is obtained. Each of the projected co-ordinates are then added to the minimum x and y co-ordinates and displayed.*

STEP-IV: IMAGE STITCHING

The next step is to stitch the projected image obtained in STEP-III with image2. Here, all the remaining pixels in image2 which are **not overlapping** with image1 are plotted.
The size of the final image is approximated with the help of the size of the warped image computed in the previous step. The resultant width of the final image is obtained by adding the widths of the two input images with the width of the warped image. The minimum x and y co-ordinates computed earlier are **added** to each of the pixels to plot the non-overlapping pixels. This performs the superimposition of image1 on image2 by translation.

NOTE: The color images are used to fill the pixel intensities in steps III and IV.