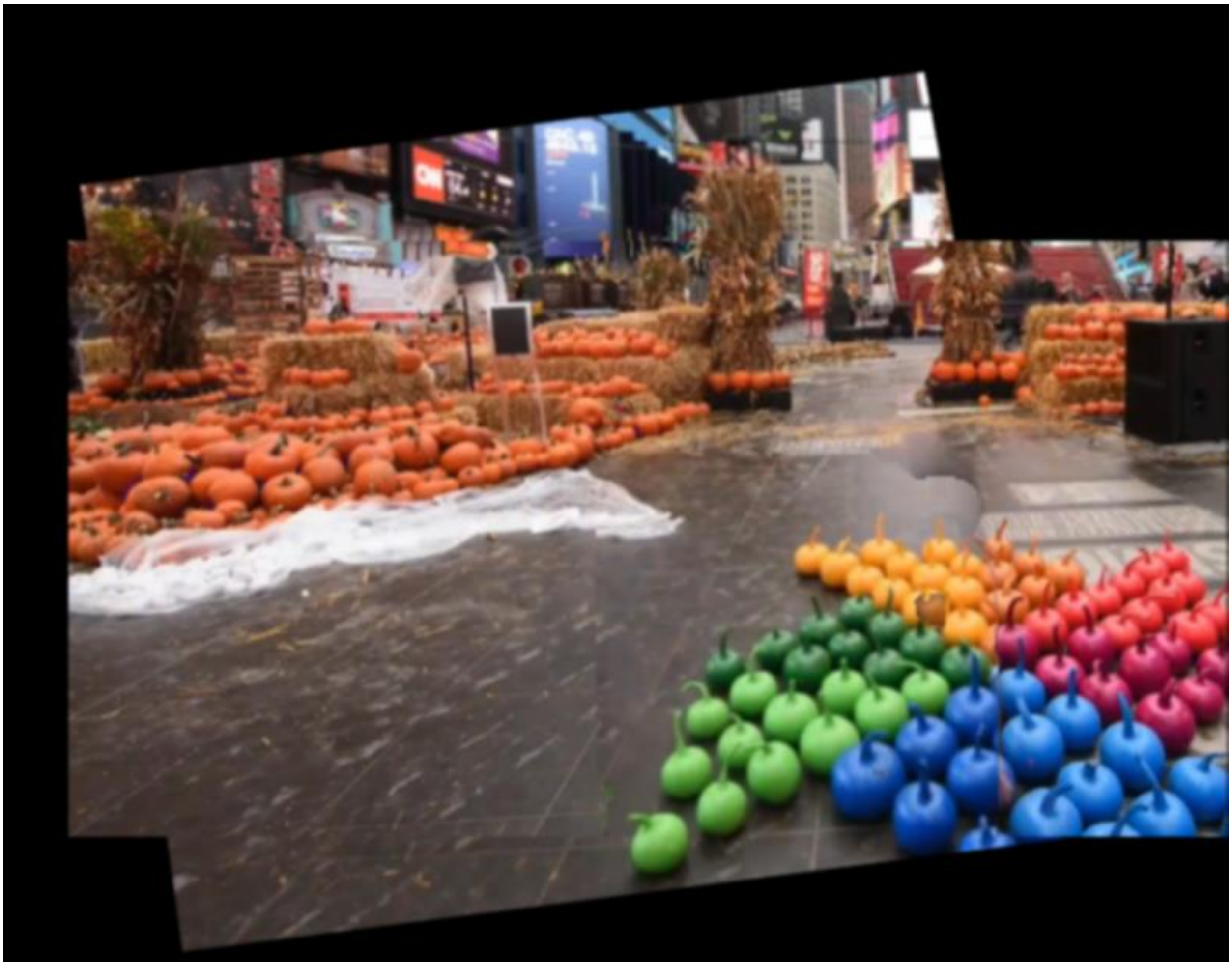
Project 2 Report

Task 1:

1. Read the both images using cv2 library and pass the images into the **“stitch_backround”** function.
2. In the **“stitch_backround”** function, call the both images and pass it to the **“get_sift_homography”** function in order to calculate **Homography.**
3. In the **“get_sift_homography”** function, we extract the keypoints and descriptors of both the images and pass it to the **“matching_keypoints”** function to calculate the Euclidean distance.
4. In the **“matching_keypoints”** function “cdist” library has been used to calculate the Euclidean distance with the help of the descriptors of both the images and obtain the matches of the images by comparing the obtained points with a threshold which is given manually.
5. After obtaining the matches, we calculate the **Homography** by using **“cv2.findHomography()”.** In this library we pass the matching points of both the images and **“cv2.RANSAC”** to find the Homography.
6. After we find the Homography, we return the Homography value to the **“stitch_backround”** function. Now, we pass the obtained value to the **“get_stitched_image”** function where we stitch the image and blend it with the help of **“cv2.grabCut”** library.
7. In the **“get_stitched_image”** function, we first give padding to the image 1 using numpy padding.
8. Now, we warp i.e., rotate the image 1 that is padded in order to blend with the image 2.
9. In order to pad the image 2, we calculate the mean of the matching points that are obtained from extracting the features and calculating the Euclidean distance of the warped image 1 and image 2.
10. After padding the image 2, the warped image 1 is stitched over the image 2.
11. After stitching, we have used **“cv2.grabCut”** library to remove the persons in the image and attaching the warped image 1 in place of the person.

Following image is the output that is obtained after blending the images:



Task 2:

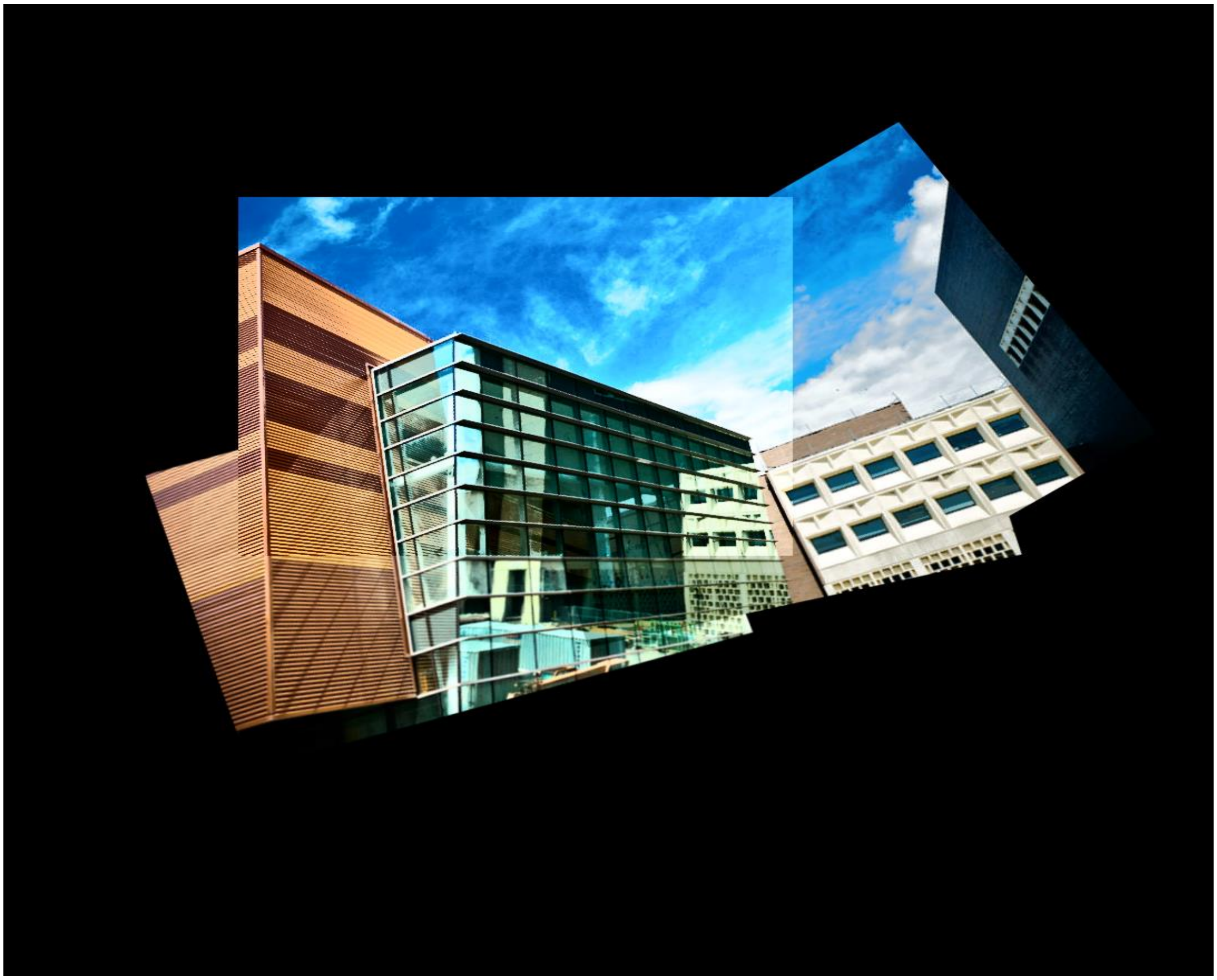
1. All the images that are required to form a panorama are stored under one name mark with a sequence of numbers tagged along.
2. Initially, I am creating a function **"stitch_2_images"** where 2 images will be passed at a time for stitching the images.
3. Initially, the first 2 images were passed here where the colors of the 2 images will be equalized first.
4. After equalizing the colors, these 2 images will be passed into the function **"get_sift_homography"** where the features of these 2 images will be extracted and matches will be found between these 2 images.

5. Based on these matches, Homography will be found using **“cv2.findHomography”**.
6. Now, the values of Homography along with the 2 images will be passed into the function **“get_stitched_image”**. In this function, the dimensions will be retrieved of the both images.
7. With these dimensions, I will fetch the dimensions of the match points by concatenating these 2 image dimensions.
8. Finally, I will warp the image 2 along with the help of match points and Homography points which will stitch the 2 images.
9. Now, with the above resulting image, this process will be repeated by sending the next image i.e., 3rd image along the result image thus forming a complete panorama.
10. For the spatial overlap matrix, I am defining a function **“spatial_overlaps_matrix”** where matches have been found for the all the combinations of the images using the **“for loops”**. Based on the matches, we are assigning values to the matrix and storing it in the json file.

Following are the results of the panorama:

Task 2 result:

Spatial Matrix: [[1. 1. 0. 1.] [1. 1. 1. 1.] [1. 1. 1. 1.] [1. 1. 1. 1.]]



Task 3 result:

Spatial Matrix: [[1. 1. 1. 0.] [1. 1. 1. 0.] [1. 1. 1. 1.] [0. 0. 1. 1.]]