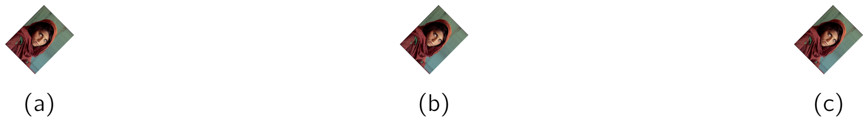
ASSIGNMENT 2: Computer Vision 792

1 Question 1



(a) (b) (c)

Figure 1: (a) blah (b) blah (c) blah

(a) (b) (c)

Figure 2: (a) blah (b) blah (c) blah

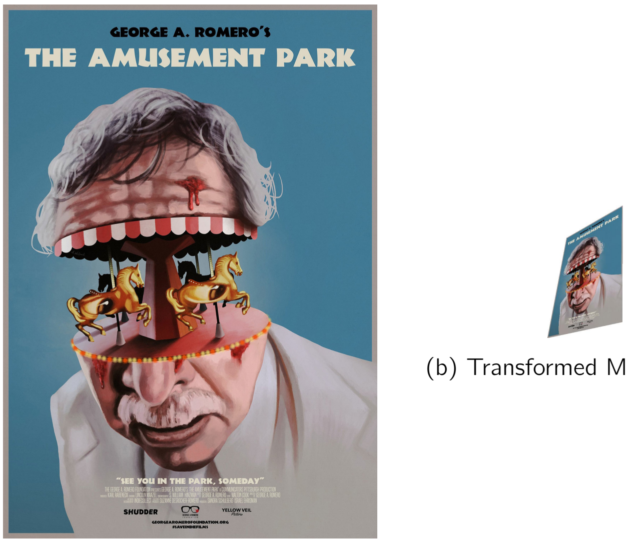
(a) (b) (c)

Figure 3: (a) blah (b) blah (c) blah

1.1 Interpretation

The `applyhomography.py` function is designed to compensate for the shift in the output image's origin. The code transforms the four corners of the input image using the homography matrix H. It then finds the minimum and maximum x and y coordinates. The size of the output image is then set to encompass all these points.

2 Question 2



(a) Original Movie Poster

(b) Transformed Movie Poster

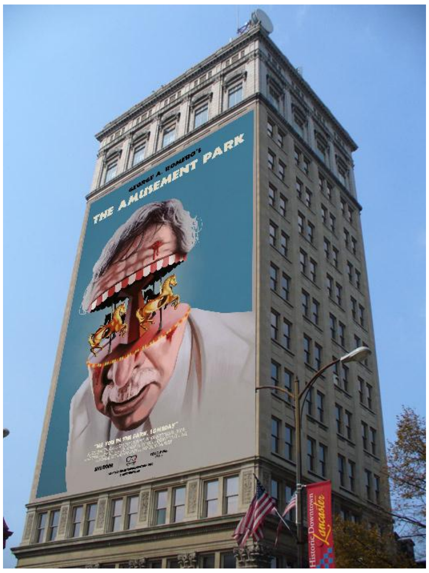


Figure 5: Building with Movie Poster

2.1 Interpretation

The transformed poster image will contain parts with no image data that we want to ignore, and the image origin may shift (why?). Explain how you addressed these two issues. The transformed poster image contains a white region where there is no image data because the transformation maps some parts of the image to areas that fall outside the original image bounds. To handle this, I created a mask: When pasting the transformed poster onto the building, the mask ensures that only the parts of the poster with actual content are pasted.

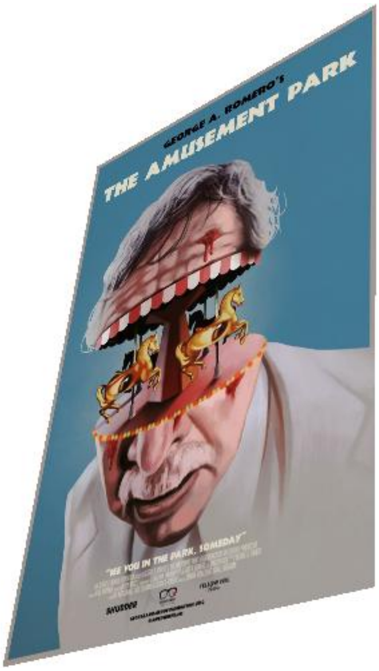
After applying the transformation, the origin of the poster shifts to a different position. This happens because the transformation can move the image in ways that push it outside the original image boundries. I compensate for this by calculating `minx` and `miny`. These represent how much the image has shifted in the negative direction from the origin `(0,0)`. In this case, the minimum x and minimum y co-ordinates shift by 35 and 124 units respectively. This offset ensures that the transformed poster aligns correctly with the intended location on the building image.

Also, if we follow the interpolation procedure in the given code that applies homographies to images, it turns out that our input (untransformed) poster image should be more-or-less similar in size to the desired output (the transformed image). If the input is much larger, we get undesired "aliasing" artefacts in the output. Experiment with this idea, and include in your report a short explanation of the issue of aliasing in image interpolation. The original image is of size 1400 x 2000 and the building image is of size 528 x 704.

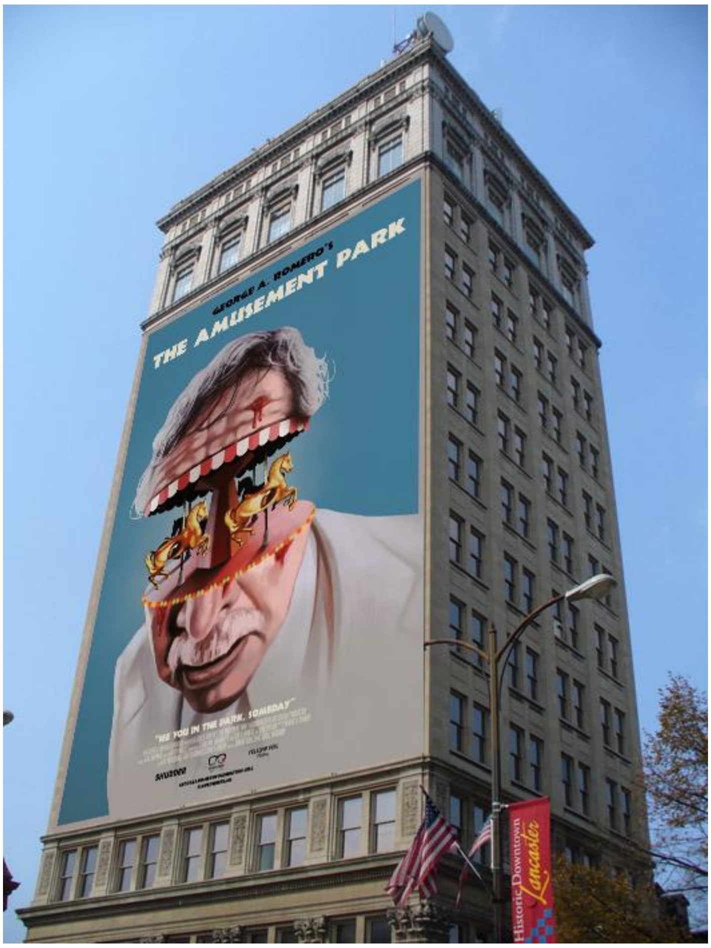
2.1.1 Case 1 : Large Input Image

As one can observe in 7b and 5. As one can observe, there are aliasing artifacts such as jagged edges and loss of fine detail. The downscaling process during the transformation fails to adequately capture the high frequency content.

2.1.2 Case 2 : Input Image of Same Size



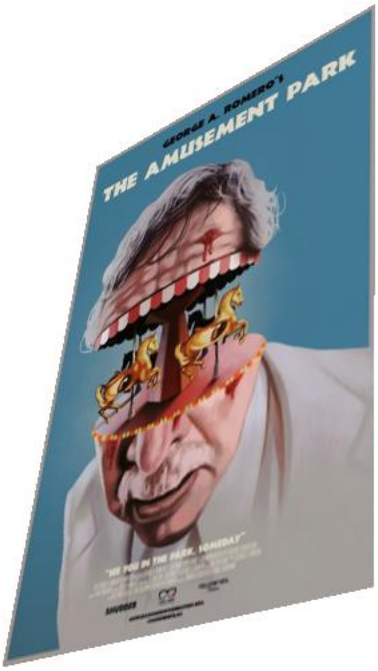
(a) Transformed Movie Poster



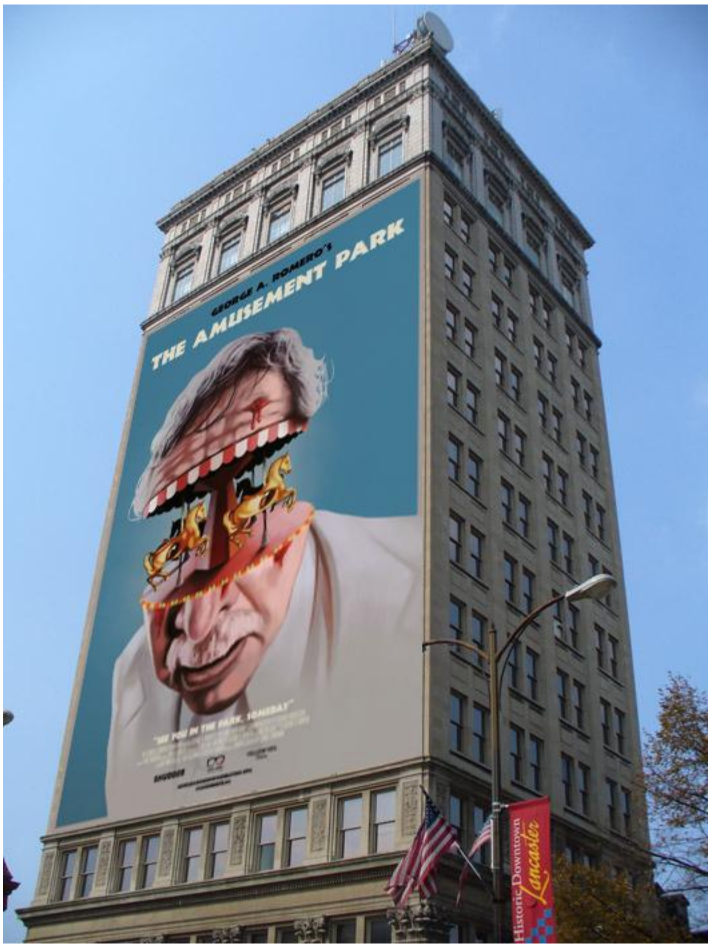
(b) Building with Movie Poster

The input image was changed to 528 x 704 to resemble the size of the output image. As one can observe, the image suffers from minimal aliasing.

2.1.3 Case 3 : Small Input Image



(a) Transformed Movie Poster



(b) Building with Movie Poster

The input image was now 250 x 350. The output poster is more blurred, but there are not as much aliasing artifacts as there were when the image was larger.