Computer Vision 1 - Assignment 4
Image Alignment and Stitching

Stephan Grzelkowski UvA-id: 10342931
Sander Brinkhuijsen UvA-id: 11305517
Sebastiaan Barneveld UvA-id: 11051108 **Whitney Mok** UvA-id: 10624767

October 9, 2019

Introduction

In this assignment, our goal is to be able to stitch two images together that have shared features, such as objects or specific patters, that might not be perfectly aligned. As a first step, identify feature points in both images and find matches between them. We perform this using the vl SIFT library. Secondly, we find an affine transformation that warps the second image so that the features are aligned. After finding the matching features and best transformation, we stitch these images together to form one 'panorama' image.

1 Image alignment

Question-1

1. and 2.

The first step to computing the affine transformation between two images, is to extract feature descriptors in each of the images, preferably descriptors robust to scale differences. Hence, we have used the VLFeat library's *vl_sift* to identify Scale-Invariant Feature Transform (SIFT) keypoints and descriptors, and let *vl_ubcmatch* find matches between descriptors in one image and in another. The matching algorithm detected 947 matches, of which 10 randomly chosen ones are visualized in Figure 1. 9 of the 10 pairs seem accurate matches, except for the one that connects the most left green point in boat1 (left image) on the bench with the most top right red point in boat2 (right image) on the trees in the background.

3.

The mismatched pixel pair illustrates how interest point matching can be very challenging for computers. Under the assumption that the matching algorithm also produces some mismatches, or 'outliers', RANSAC can be helpful for finding transformation parameters. Through iterative computation of transformation parameters based on a random subset of matching points and determining which parameters are best considering the inliers score, we can obtain the best transformation parameters. Figure 2, displaying boat1 on the left and boat2 on the right, shows that the lines between boat1's transformed points (right) and the original points in boat1 (left) are parallel, and that the red points do hover over points in boat2 that correspond to their green counterpart in boat1 (left). RANSAC's good performance is further illustrated in Figure 3: the transformation of boat1 (left) is very similar to the boat2 image (not shown here). Naturally, performing the transformation on boat2 itself (right) using the same parameters led to even further counterclockwise rotation.

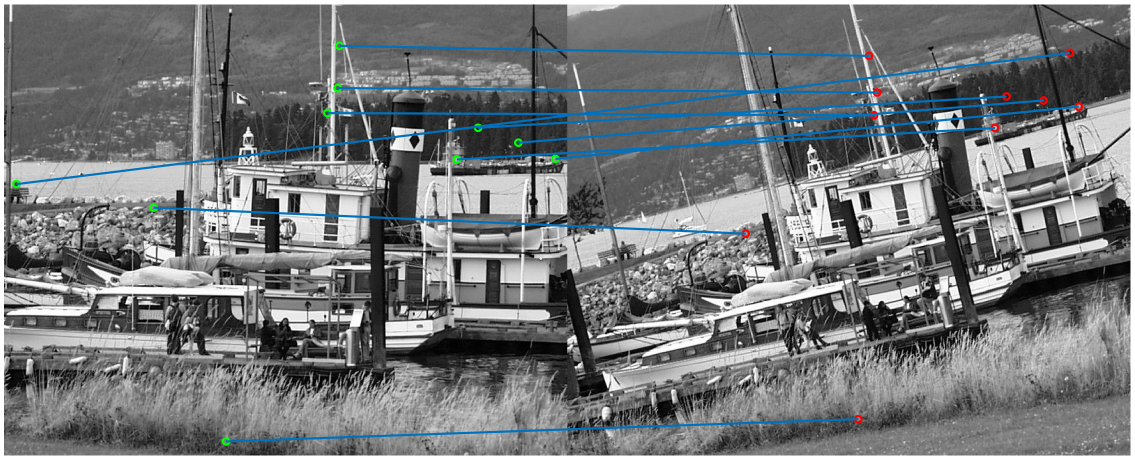


Figure 1: A random subset of matching points between boat1 and boat2.

Question-2

1.

The minimum number of points that we need to solve an affine transformation is 3. This is because an affine transformation can rotate, scale, and translate the original image, but the lines remain parallel. This means that the relative angle between two lines remains the same. We can also show that 3 points are needed by using the formulation of the affine transformation in the exercise.

$$\begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

In the equation there are six unknowns, m_1, m_2, m_3, m_4, t_1 and t_2. Each (x, y) coordinate provides us with two equations, meaning that we would need 3 (x, y) coordinates in order to solve this system.

2.

In practice we have found that 10 runs provide us with a good enough chance for a result with sufficient inliers for a accurate transformation. This almost always gives an accurate result with a small computation cost, whereas 5 runs still sometimes results in inaccurate transformations with few benefits in computation time.

2 Image stitching

Question-1

1. and 2.

We again used RANSAC to align right.jpg with left.jpg. Subsequently, we calculated the matches from the aligned right image and the left image to determine where the images overlap.

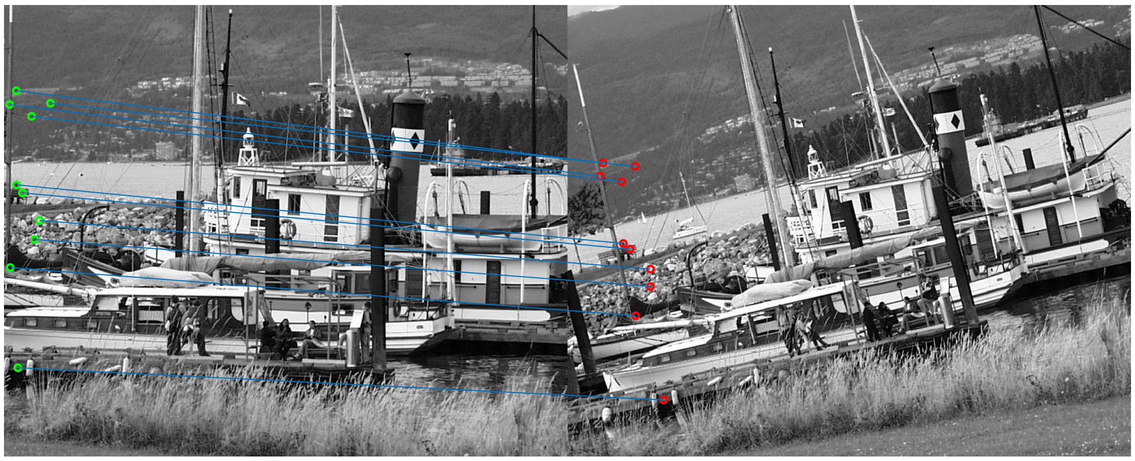


Figure 2: The transformations of ten points in boat1 shown above boat2.

Now we create an empty stitched image where we will insert both images. To calculate the size of the stitched image we take the corners of right.jpg and apply the transformation matrix to them. Next, we use the matches to determine the difference in their coordinates and insert the left image in the stitched image and overlay the right on top of it.

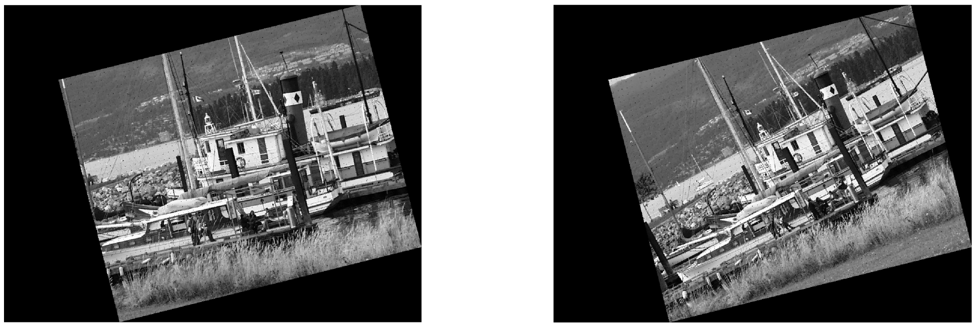
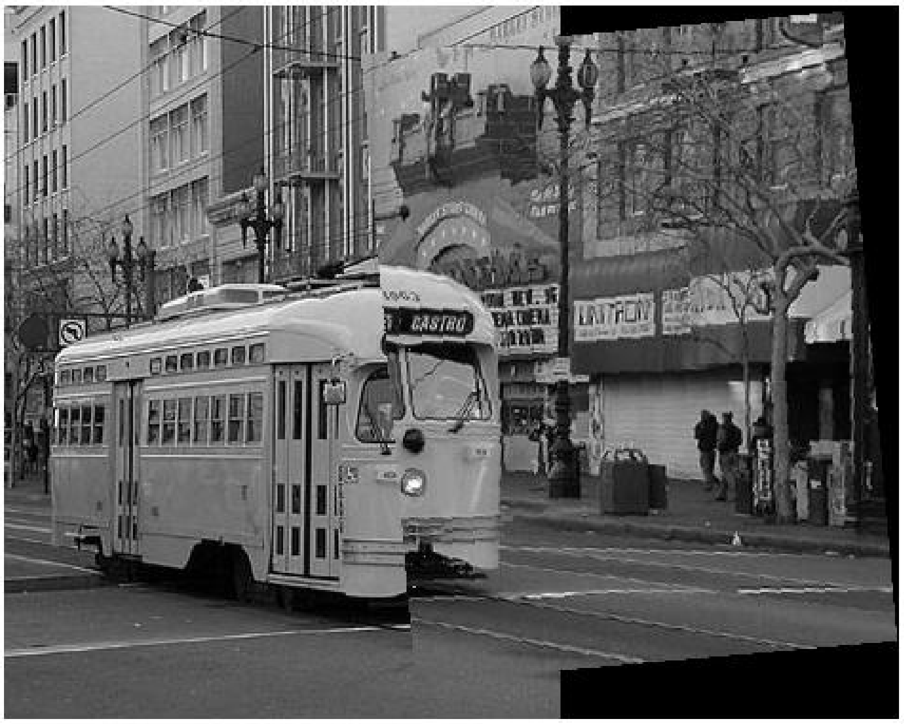


Figure 3: Transformation of boat1 (left) and boat2 (right) using the best transformation parameters computed for a transformation from boat1 to boat2 using RANSAC with a random sample size of 10 in each of the 10 iterations. The continuous transformations were fitted into the new, discrete image space by the nearest neighbour approach.



(a) The result of stitching the images

The result of stitching is shown in Figure 4a, which shows that the black border of the transformed right image is inserted on top of the left image. This is not a problem, however, as the resulting stitched image still clearly proves that the implementation is successful in stitching similar images.
The result is not perfect as the y-axis does not line up exactly, we believe that this is due to the random nature of RANSAC not providing us with the exact transformation coordinates.

Conclusion

In this report we discussed our implementation of the RANSAC algorithm and its use in stitching images. First we used Matlab's build-in SIFT algorithm to find matching keypoints between two similar images. Then, using our own implementation of the RANSAC algorithm we determined the affine transformation parameters to convert the first image into the second image for question 1.
Using the same technique for question two determined the parameters to stitch two images with different perspectives together, overlapping them on top of each other to provide a new image with more information than the original.