Assignment 4
Image Alignment & Stitching

Nicole Ferreira Silverio
10521933

Cor Zuurmond
10580395

1 Introduction

For the course Computer Vision 1 at the Universiteit van Amsterdam (UvA) weekly assignments regarding computer vision have to be submitted. This week the assignment was on 'Image Alignment & Stitching'.

For this assignment two different subjects are reviewed; Image Alignment and Stitching, where image alignment is needed to stitch images together. In the coming sections of the report each partial subject will be discussed in more detail by answering the questions that were provided in the assignment. Firstly, in section 2 Image Alignment will be discussed and secondly in section 3 Stitching will be elaborated.

2 Image Alignment

For the first part of the exercise image alignment is implemented following these steps; first, SIFT keypoints in both images are detected; secondly, keypoints in the images are matched; thirdly, iteratively a random subset of matching keypoints are selected to find the best transformation vector $\boldsymbol{x}$, using the RANSAC algorithm; lastly the image is transformed.

Keypoint are found and matched with two functions from the VLfeat toolbox. In the first step, *vl_sift* is used to compute the keypoint frames and descriptors. In the second step, *vl_ubcmatch* finds matching keypoints.

RANSAC uses these matching keypoints to find the most accurate transformation vector $\boldsymbol{x}$. Some keypoint pairs might be incorrectly matched. Incorrect pairs can offset the transformation vector a lot from the 'true' transformation. To find a transformation vector most closely to the 'true' transformation, n times a random subset of size p of the matched keypoints is drawn to calculate the transformation vector $\boldsymbol{x}$. The transformation vector is considered to be most accurate if it returns the most 'inliers'. An inlier is a transformed keypoint that is within a r pixels from its match. By default $n = 10$, $p = 50$ and $r = 10$. Below the definitions of the transformation vector $\boldsymbol{x}$ is given:

$$
\begin{aligned}
\begin{bmatrix} x' \\ y' \end{bmatrix} &= \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \\
\boldsymbol{A} &= \begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix} \\
\boldsymbol{x} &= \begin{bmatrix} m_1 & m_2 & m_3 & m_4 & t_1 & t_2 \end{bmatrix}^T \\
\boldsymbol{b} &= \begin{bmatrix} x' \\ y' \end{bmatrix}
\end{aligned}
$$

where x' and y' are the xy-coordinates of a keypoint in one image, and x and y the xy-coordinates of its match in the other image. With the transformation vector the image can be transformed with

the formula above. The value of each pixel in the transformed image is found by transforming the location of a pixel back to the original image and finding the nearest pixel value.

2.1 Question 1

The image in figure 1 are transformed.

Figure 1: Two images of a boat.

In figure 2 the matched keypoints (green circles) in the two image are connected with yellow lines.

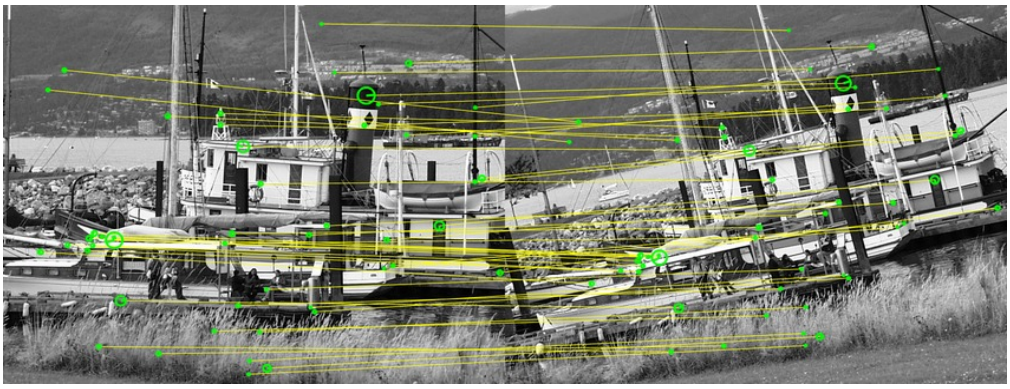

Figure 2: Matched key points between the two boat images.

In figure 3 the left boat image is transformed to match the right boat image.

Figure 3: Transforming one boat image to the other.

In figure 4 the right boat image is transformed to match the left boat image.

Figure 4: Transforming the boat images the other way around.

For comparison the images are also transformed using built-in functions ‘affine2d’ and ‘imwarp’. The results look identical to the transformation created with our own implementation.

Figure 5: Transforming images using Matlabs built-in functions.

2.2 Question 2

There are six unknowns in the affine transformation. Per match we get two equations to solve, therefore we need at least three matches to solve the affine transformation.

To find good transformation parameters, i.e. parameters that transform at least 90% of the 50 transformed points as inliers, for the boat images on average 3.6 iterations of the RANSAC algorithm are needed, with a default threshold of 1.5 for the ‘vl_ubcmatch’ function, $r = 10$ and $p = 50$.

3 Stitching

Stitching extends on image alignment. First one image is transformed to match the alignment of the other image, as explained above. Secondly the images are stitched. To stitch, the relative offset between the images is needed. The corners of the transformed image are used to find this offset. The average of the left and right image is taken where the images overlap. The images in figure 6 are stitched, see figure 7.

Figure 6: The left and right image.

Figure 7: The stitched image.

4 Conclusion

Image alignment is implemented by finding the most accurate transformation vector using matching keypoints. Keypoints are found and matched with functions from the VLfeat toobox. RANSAC and transforming an image is implemented by us. Results of our own implementation are visually identical as the results from the built-in functions of MATLAB to transform an image. Having two left and right images aligned, they are stitched together in the last part.