Panoramic image stitching project
TNM089

Isak Engström
isaen688@student.liu.se

Henrik Nilsson
henni317@student.liu.se

Department of Science and Technology
SE-581 83 Linköping, Sweden

Abstract

When creating a panorama image, the most basic approach is to simply stitch the images together. This produces a poor result in most cases, and the preferred approach is instead to use some method for blending the images together, making for a smoother seam.

This report aims at describing the implementation of a stitcher, followed by comparing the blending methods to one another, more specifically the feather blending method and the multi-band blending method. Our result has shown that neither of the methods is superior in every case, and that the preferred method to use in each case depends on a number of variables. We present a set of factors that can help the reader to choose which blending method to use when stitching a panorama together.

1. Introduction

This report details the implementation and comparison of different methods used for blending two or more images into resulting panorama images. The algorithm used for the panoramic image stitching is the one presented by Brown and Lowe in the paper *Automatic Panoramic Stitching using Invariant Features* [4], where blending is the final step of the algorithm. The blending methods that have been compared are feather blending, and multi-band blending with a differing number of bands.

2. Systems architecture

The implementations will be written in the programming languages *Python* (version 3.9) [2] and *C++* (version 17) [6]. The initial stitcher will be implemented in Python. This version of the stitcher will include the ability to crop the final panoramas, as to remove the black borders otherwise created. The C++ version will be used for the evaluation of the blending methods, as *pointers* are a useful tool for making small changes in the stitcher pipeline.

The languages will be combined with the computer vision library *OpenCV* (version 4.4) [1], which is written in optimized *C* and C++ [6]. The library is designed to perform fast calculations, partially because of its use of multi-core processing.

3. Datasets and evaluation protocol

The data used in the project consist of different image datasets. The *Rooftop dataset* [9] is provided by *MareArt's blog* and contain five overlapping, 320 x 240 pixel images. These can be seen in Figure 1. Another dataset, referred to as the *Campus dataset*, consist of six overlapping images. These were taken by a smartphone camera and are 4032 x 3024 pixels in size. Part of the Campus dataset is displayed in Figure 2. The images in these datasets were taken by hand without using a tripod. However, the cameras were, as best as humanly possible, rotated around their optical centre whilst taking the images.

Figure 1: The images of the Rooftop dataset

Figure 2: Three images from the Campus dataset

The speed of each blending method will be evaluated by comparing them to each other. To evaluate the quality of the stitching (and more specifically the blending), the human visual system is used. As presented by Dissanayake et al. [7] and Boutellier et al. [3], there exists measurements to evaluate the qualitative stitching results. However, implementing these are outside the scope of this project.

4. Methods

The following sections will present the theoretical background of the stitching algorithm by Brown and Lowe [4], followed by a description of how the blending methods are evaluated.

4.1. Feature extraction

The first step of the algorithm is to extract features in each of the images. This is done through the method known as *Scale-invariant feature transform* (SIFT) [8]. For each image, one or more downscaled Gaussian blurred versions are created and put in a stack. By subtracting each of these blurred images from the previous image in the stack, a *Difference of Gaussians* (DoG) is obtained. Every pixel in each of the DoG images are then compared to its surrounding 8 pixels and the 9 pixels of the same area in the two closest DoG images. If the pixel value is higher than all of the neighboring pixels, it is a local extrema. If the pixel value is lower than all of the neighboring pixels, it is a local minima. In both of these cases, it is kept as a potential feature. Each feature has a magnitude depending on the size of the DoG image that it was extracted from. Furthermore, by subtracting the orientation of each feature from the orientation of its surrounding pixels, both rotation and scale independence is achieved, making the method invariant to changes in rotation and scale. The features are then stored in a descriptor vector.

Although SIFT is invariant changes in scale and rotation, the matching is still sensitive to variance in illumination. Change in illumination can firstly occur when the lighting is changed. The camera can also be susceptible to change in illumination when it is rotated, and different parts of the motive is displayed. Invariance is acquired by normalising the descriptor vector using gradients [4].

4.2. Image matching

Once the features have been extracted for all the images, the next step is to find the matching features in between them. Assuming that the camera has a fixed position, the images that depict the same stationary motives (i.e. overlapping images) contain many of the same feature points. *Feature space* is used to relate the feature points to each other. The feature points that represent the same feature in different images, show up close to each other in feature space. By using the *k nearest neighbour* algorithm on the feature space, it is possible to find the images that have many matching features in between them. These are the potential image matches that should be combined into a panorama [4].

A feature point that show up in two images can be referred to as a *correspondence*. The goal of image matching is to map an image to the other by using these correspondences, so that they are overlapped in the correct manner. This is done using a *(Planar) homography* [10], followed by verification step to determine if the match is correct [4].

4.2.1 Homography / Overlap calculation

For a pair of images containing correspondences, one of the feature points can be overlapped at its corresponding point in the other image. However, this method is not sufficient to create panoramas. The images can be distorted depending on how the camera is angled, resulting in two feature points not being equidistant to each other, compared to their correspondences. Instead, the homography is used, which can map all points between two planes. This is done by multiplying the homogeneous coordinates of a point with the homography matrix, as seen in Eq. 1 [10].

$$p_2 = H * p_1 \tag{1}$$

Where H is a $3x3$ matrix of scalar values:

$$H = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{pmatrix}$$

and p_1, p_2 are homogeneous coordinates:

$$p_x = \begin{pmatrix} u_x \\ v_x \\ 1 \end{pmatrix}, \quad x = 1, 2$$

The homography has eight *degrees of freedom* (DoF) that need to be solved for before all the image points can be mapped. This is also represented by the eight values in H. To fully solve this, at least four non-collinear correspondences are needed [4], [10]. After H has been calculated, it can be used to warp the images into the correct overlap, which accounts for potential distortions in between the images [10].

Some feature points can be unreliable and should not be used when solving for H. This inaccurate data may occur due to noise. However, it can also occur when an object which produces a feature point has moved in the scene in between taking the photos. This unreliable data that fit a model with large error is referred to as *outliers*, whilst data that fits a model with small error is referred to as *inliers* [10].

To determine the homography in practice, a method called *Random sample consensus* (RANSAC) is used.

RANSAC uses a set of random correspondences and estimates H for a number of trials. The trial which leads to the greatest number of inliers is chosen as the most appropriate estimation of H. In the paper by Brown and Lowe, 500 trials are conducted with four random correspondences each. This method is robust, as the probability of *not* finding the correct homography is approximately 10^{-14} [4].

4.2.2 Image match verification

The last step of the image matching is to verify if the pairwise matching is correct. The model which performs this verification looks at the probability of a set of inliers and outliers being generated by a true image match. This way, false image matches can be eliminated [4].

4.3. Connect matches

From the image matching step, potential pairwise matches have been established. Images that have no connections will not be part of any panorama. Further, all images that should belong to a single panorama do not need to be directly connected to each other, it is enough to be linked through other connections. If there are clusters of connected images without a link in between the clusters, these are considered as different panoramas [4].

What remains before the panoramas are complete is to perform the stitching. This, in itself, consist of two steps:

- Overlapping the images with *bundle adjustment* and the previously created homographies.
- Blending the images along the seams.

4.4. Bundle adjustment

The images that belong to a panorama are not added all at once. Instead, bundle adjustment is used, which is an iterative process of adding one match to the panorama at a time. The method adjusts the camera parameters and feature points simultaneously at each step, as to avoid accumulated errors which occur when homographies are stacked [4]. However, bundle adjustment is sensitive to outliers, these should therefore have been removed beforehand [10].

4.5. Blending

The last step of the algorithm is the blending. In the original article by Brown and Lowe [4], multi-band blending is used as the only approach. This method has been examined more closely by looking at the result when varying the amount of bands, and has also been compared to feather blending to find the optimal blending method for each situation.

To more easily be able to evaluate the blending methods, the Rooftop data will be manipulated. The middle image, which can be seen in Figure 1, will have its light intensity lowered. A comparison between the light intensity of the original and the manipulated version can be seen in Figure 3.

Figure 3: The middle image of the Rooftop dataset, with original intensity to the left and lowered intensity to the right.

4.5.1 Feather blending

Feather blending is one of the simpler methods that can be used. When blending two images, an area contained in both of the images is blended linearly using both images. This straight-forward approach tends to work well when the areas are almost identical in both images. When they differ even slightly, ghosting artifacts tend to appear where segments of both images are visible, especially when the area that is blended over increases in size. An example of this can be seen in Figure 4, where two completely different images have been blended. This clearly shows overlapping segments from both stitched images in the centre of the stitched image.

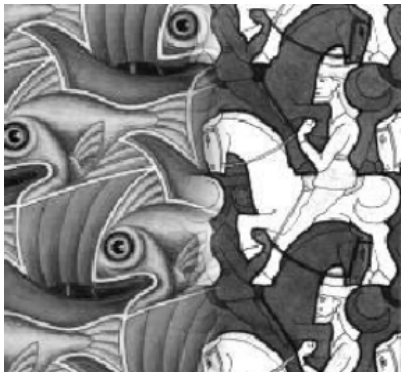

Figure 4: A feather blend of two images with ghosting artifacts in the centre.

4.5.2 Multi-band blending

Multi-band blending, as presented by Burt and Adelson [5] is a more advanced blending method, where the blending is

performed on multiple frequency spectra. The images to be blended are each split into a number of new images, matching the number of bands. This is done by applying consecutive Gaussian blur to the image, similarly to what was done in the previously mentioned feature extraction. This results in images that contain the lower frequencies while removing the higher ones. By subtracting an image containing a lower frequency spectrum from one with a higher frequency spectrum, a resulting image that only contains the higher frequency spectrum which sets them apart is created.

This approach is repeated for all of the different frequency spectra, and the result is a stack of images containing the exact same information as in the original image, where each stack contains a certain frequency spectrum. Once this has been achieved, each frequency spectrum of an image is blended with the same frequency spectrum of the other image over an overlapping area that increases in size for the lower frequency spectra. When all frequencies have been blended, the resulting stitched images containing the frequency spectra can simply be added together to achieve the resulting panorama image.

5. Results

This section will show panoramas that have been stitched from images with both feather blending and multi-band blending with differing amounts of bands. As mentioned in section 4.5, the middle image of the rooftop dataset has had its light intensity greatly reduced to make the differences between the blending methods more apparent. No significant difference in speed is noticed between the blending methods, not even when using different number of bands for the multi-band blender.

Figure 5: The Rooftop dataset images stitched using feather blending and reduced light intensity in the middle image.

6. Discussion

Comparing the different results that have been achieved using the Rooftop dataset with different blending approaches, the panorama in Figure 7 gives the most natural looking result. Using too few bands, as seen in Figure 6, tend to result in rougher edges because of not having enough frequency spectra to blend over. Using too many bands, as seen in Figure 8, will not leave any rough edges. That does not mean it gives a good result, as the resulting image will become more blurred, which takes some of the detail out of highly detailed areas. It also noticeably changes the colour of the image, due to increasingly bigger areas being blended as the frequency spectrum becomes lower, as explained in

Figure 6: The Rooftop dataset images stitched using multi-band blending with 2 bands and reduced light intensity in the middle image.

Figure 7: The Rooftop dataset images stitched using multi-band blending with 5 bands and reduced light intensity in the middle image.

Figure 8: The Rooftop dataset images stitched using multi-band blending with 9 bands and reduced light intensity in the middle image.

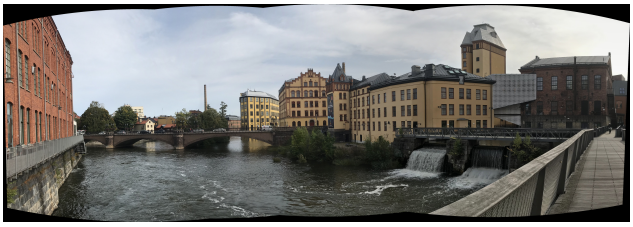

Figure 9: The Campus dataset images stitched using feather blending.

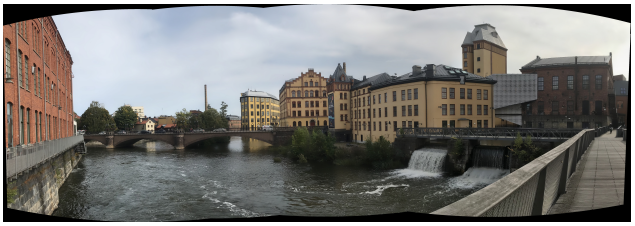

Figure 10: The Campus dataset images stitched using multi-band blending with 5 bands.

section 4.5.2. Using feather blending does not make for a satisfactory result with the Rooftop dataset images due to the rough stitching done at the seams. However, highly detailed areas of the resulting image undoubtedly looks better, compared to when using multi-band blending.

This blurriness is something that can not be avoided when using multi-band blending, especially as the amount of bands increase, which is evidently apparent if comparing Figure 9 and Figure 10. Although most of the areas look good in both images, certain areas lack consistent detail when using the multi-band blending method, such as the clouds in the sky. With this area consisting of around a third of the entire image, this is a situation where using feather blending will give a more satisfying result.

There is a slight difference in the execution time for the stitcher when using the different methods. However, since the blending is only a part of the entire algorithm for creating the panorama image, this conducted study has shown that the choice of method makes a marginal difference on the total execution time. It is therefore not something that needs to be taken too much into consideration when choosing an appropriate blending method.

With neither of these methods being optimal for every situation, and with the difference in image quality from differing the amount of bands when using multi-band blending, it can be hard for someone without the proper knowledge of what sets the methods apart to make an educated guess of which method to use. One way to handle this would be to automatically select a method depending on the values of the variables previously mentioned, and in the case of multi-band blending set the number of bands depending on the image size. Such an application is unfortunately outside of the scope of this course, however, it would be an interesting approach to further the work.

7. Conclusions

To conclude, feather blending can be a good approach given the proper conditions. Highly detailed images shot in a static environment from the same camera position will likely work best with feather blending, as the level of detail will be kept, and no ghosting artifacts will appear. In less controllable environments, where the common scene inevitably will vary between the images, multi-band blending will most likely produce a better result. However, other aspects such as the level of detail in the image and differences in light intensity also play an important role in selecting the best approach to use for a given set of images. To select the optimal blending method, it is therefore preferable to manually choose a method depending on the images to be blended, or automatically choose a method depending on the different aspects that exist in the images. The blending is a marginal part of the stitching and the speed of each blending method does not need to be taken too much into consideration.

References

[1] Opencv. Version: 4.4, 2020. https://opencv.org/.

[2] Python. Version: 3.9, 2020. https://docs.python.org/3/.

[3] J. Boutellier, O. Silvén, L. Korhonen, and M. Tico. Evaluating stitching quality. In *Proceedings of the Second International Conference on Computer Vision Theory and Applications - Volume 1: VISAPP,*, pages 10–17. INSTICC, SciTePress, 2007.

[4] M. Brown and D. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, 74(1):59–73, 2007.

[5] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Trans. Graph.*, 2(4):217–236, Oct. 1983.

[6] cplusplus.com. C++ documentation. 2020. https://www.cplusplus.com/.

[7] V. Dissanayake, S. Herath, S. Rasnayaka, S. Seneviratne, R. Vidanaarachchi, and C. Gamage. Quantitative and qualitative evaluation of performance and robustness of image stitching algorithms. pages 1–6, 11 2015.

[8] D. Lowe. Distinctive image features from scale-invariant keypoints. 2004.

[9] MareArts. Mare's computer vision study. 2013. http://study.marearts.com/2013/11/opencv-stitching-example-stitcher-class.html.

[10] K. Nordberg. Introduction to representations and estimation in geometry. Version: 0.40, September 18, 2018. http://liu.diva-portal.org/smash/get/diva2:1136229/FULLTEXT03.pdf.