Ukrainian Catholic University

Faculty of Applied Sciences

IT & Business Analytics Programme

Image Stitching

Linear Algebra Second Project Report

Authors:
Alina Voronina
Viktoriia Maksymiuk
Yuliia Maksymiuk

13 May 2022

1 General overview of the problem

The Image Stitching project is a work in the field of computer vision. Image stitching algorithms are widely used in various spheres: panorama photography, space imaging, and satellite image data processing. The primary purpose is to represent several images as a single one, solving the problem of limited resolution and frame dimensions. The obtained "stitched" image is easier to process and is more informative than tens or hundreds of smaller shots. Therefore, the development of such an algorithm is essential for computer vision. In addition, it is interesting to try to create one's own image stitching algorithm from scratch using knowledge of linear algebra.

The aim is to implement an algorithm that composites multiple overlapping images captured from different viewing positions into one natural-looking panorama, using linear algebra principles and algorithms. It will help obtain a full-range view in a single photo, excluding the necessity of manual stitching in photo editors. The final goal is to create a program that receives some images of an object/surface/scene as an input and outputs a single picture of that.

Why would one want to use the image stitching algorithm? Imagine visiting the Burj Khalifa - the highest building in the world. With the regular camera and the smartphone, one will not be able to capture the tower in detail, maintaining a high resolution and pretty detailed view. One would most probably use a panorama mode or try to photoshop several images into a single one to post on social media to get likes from friends. Cropped Burj Khalifa will not look as impressive on the Instagram page **:)** Another situation is taking pictures with a large group of people. If one is lucky enough to have friends, they most likely had a situation where they wanted to take a picture of them all. In this case, no one wants to be cropped or left outside of the frame.

As photoshop tools may appear complicated to use for a regular person, the image stitching program will save them. Using Linear Algebra techniques and existing libraries, the project's algorithm will solve the problem of several images composition very precisely.

2 Work review

There are three main steps in implementing the image stitching algorithm. Many approaches are proposed to perform each of them, every one having its fors and againsts.

A Keypoint detection and local invariant descriptors extraction

To be stitched together, the images need to have some common elements. The view of these elements may differ, depending on the angle, scaling, rotation, or spatial position. The most straightforward technique to use is the algorithm called **Harris Corners**. It directly takes the corner score's differential into account concerning direction [1], and it happens to be invariant to rotation. Nevertheless, the input images may differ not only by rotation angle but also by scale. The Harris Corner detector (being not scale-invariant) will not work in such a case, which is a clear disadvantage.

Among the other techniques that may be used in keypoint detection are the following: scale-invariant features transform (**SIFT**), speeded up robust features (**SURF**), **Oriented fast**, and rotated BRIEF (**ORB**), and so on. SURF is a little similar to SIFT, and it is slightly quicker than the latter one. However, SURF is not stable to rotation, and it does not work correctly with illumination [2]. In this project, the classic SIFT will be implemented, as it is a more reliable method used in many works. This technique is both rotation and scale-invariant, although it happens to be slightly slower than the alternatives.

B Feature matching

After obtaining features and descriptors for the images, the next required step is to establish preliminary matches between these photos. For this task, two matching methods can be considered: **Brute Force Matcher** and **K-Nearest Neighbors** (KNN). Brute Force calculates the distance between two points using Euclidean distance. Thus, for each descriptor in image 1, it returns the nearest descriptor (single best match) in image 2 and vice versa. However, this is not very efficient because there will be many false positives, resulting in the homography matrix not being correct.

Contrastingly, KNN (*K-Nearest Neighbors*) is useful when considering more than one candidate match, as this method returns K best matches for a given descriptor. One of the main advantages of KNN over Brute-Force is the possibility of manipulating the matches obtained by implementing a ratio test. It is useful when a correct match has a larger distance than an incorrect one or when many similar descriptors of repetitive features in these images are present.

Nevertheless, the KNN-matching method can be very effective, but it requires storing the whole training set; therefore, it may use much memory and be relatively slow. Another weakness is selecting the optimal K value to achieve the maximum accuracy of the model, which is challenging. However, K=2 is optimal for image stitching *when performing the ratio test.*

C Homography estimation with RANSAC algorithm

The most exciting part of creating panoramas is that the two images are not that of a plane but a 3D scene when a person takes them by rotating the camera about its optical axis. To align images of a panorama, at the heart of this technique is a simple invertible 3×3 matrix called **Homography**.

An alternative to using homographs or 3D motions to align images is first to warp the images into **cylindrical coordinates** and then use a pure translational model to align them [3]. Unfortunately, this only works if the camera takes all images at the *same level or with a known inclination angle.*

Pictures can also be projected on a **spherical surface**, which is helpful if the final

panorama *involves an entire sphere or hemisphere view*, not just a cylindrical strip [4]. Algorithms for merging cylindrical images are often used when the camera is known to be level and rotates only around the vertical axis. In such conditions, a **purely horizontal translation** connects images with different rotations. Since homography is **very sensitive** to the quality of data we pass to it, there is a need for an algorithm to filter irrelevant points from the distribution. **RANSAC** is the best-suited candidate for that job.

The most famous RANSAC alternatives are **Least Squares Fit** and **Hough transform**. Both of these procedures are similar to RANdom SAmple Consensus algorithm since they are robust to noise. However, the Least Squares Fit is a simple closed-form solution in addition to the not resistance to outliers.

The Hough transform can fit multiple models; however, its weakness relates to the works for a few parameters (typically 1-4). A RANdom SAmple Consensus algorithm works with a moderate number of parameters (1-8). Even though both of the techniques are robust to outliers and noise, RANSAC is still the best technique to use while working with estimating a homography.

However, there are **drawbacks** too. First of all, when the number of measurements is quite large, RANSAC works slower. That is why there are modifications of RANSAC, which can significantly speed up its performance (*Preemptive RANSAC*). Secondly, in another variant on RANSAC called PROSAC (*PROgressive SAmple Consensus*), random samples are initially added from the most "confident" matches, thereby speeding up the process of finding a (statistically) likely good set of inliers. There are a lot of improved algorithms that can are more useful in end-to-end training of feature detection and matching pipelines. Nevertheless, many state-of-the-art pipelines still rely on methods that **stood the test of time**, such as SIFT or RANSAC, which authors implement for the project.

3 Methodology and theoretical part behind it

A Keypoint detection and local invariant descriptors extraction

There are four core steps for SIFT algorithms: scale-space extrema detection, keypoint localization, orientation assignment, and keypoint descriptor. Firstly, the image pyramid from each image is generated, and the Difference of Gaussian (DoG) is calculated. Once the DoG is found, images are searched for local minima and maxima over scale and space. For instance, 1 pixel in an image is compared with its eight neighbors and 9 pixels in the next scale and 9 pixels in previous scales [5]. Local extremas are potential keypoints that are also filtered for relevance by comparing their intensity with the chosen threshold. Keypoints also must not be located at the edges of the image, as they may vanish at the second image one wants to stitch it to. The edge keypoints may be detected via eigenvalues (Harris detector), as shown below:

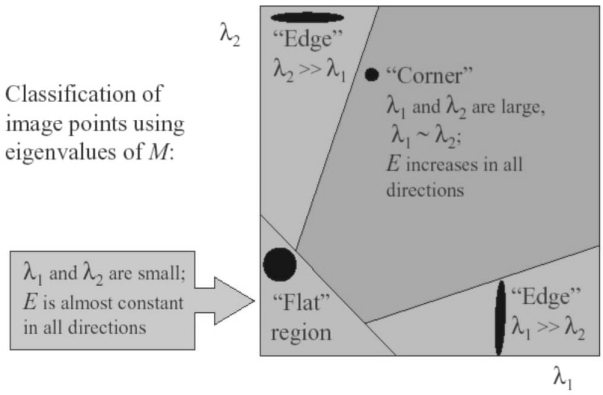

Figure 1. *Classification via Eigenvalues*

Consider D, which is the result of DoG method:

$$D(z_0 + z) \approx= D(z_0) + (\frac{\partial D}{\partial^2 z}|_{z0})^T z + \frac{1}{2} z^T (\frac{\partial D}{\partial z^2}|_{z0}) z$$

It is the 2nd order Taylor series expansion. The Hessian (matrix of derivatives) H in x and y is as follows:

$$\mathrm{H} = \begin{bmatrix} D_{xx} D_{xy} \\ D_{xy} D_{yy} \end{bmatrix}$$

It is evaluated for every feature point, and the ratio of the eigenvalues 1 and 2 of H (they correspond to the principal curvatures) is compared to a threshold ratio r:

$$\frac{Tr^2(H)}{Det(H)} = \frac{(\lambda_1 + \lambda_2)^2}{\lambda_1 \lambda_2} < \frac{(r+1)^2}{r}$$

High ratio points are considered to be edge points and are rejected. The last steps are keypoint orientation assignment and keypoint description.

B Feature matching

Implementing the K-Nearest Neighbours Matcher is relatively simple because the central concept of LA underlying it is the Euclidean distance. For each descriptor from image1, one finds K nearest neighbors from image2 after calculating the Euclidean distance between two points in the Euclidean space.

$$d\,(x, y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$

The optimal value of K for this algorithm is 2 to keep the two best descriptors for each descriptor in both images (cross-validation) and perform a Lowe's ratio test. This method checks whether the two distances are different enough to determine either to use the keypoint or to exclude it from further calculations, thus finding correct feature matches.[6]

C Homography estimation with RANSAC algorithm

The abbreviation stands for *RANdom SAmple Consensus*, an algorithm proposed in **1981** as a method to estimate the parameters of a particular model starting from a set of data contaminated by *large amounts of outliers*. The **RANSAC** algorithm calculates a whole estimation of the homography matrix based on a threshold value (ε). Finally, it chooses the homography that **maximizes the number of inliers**. This technique finds the best perspective transformation **H** between the source and destination planes

$$s_i \begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} \sim H \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

so that the ***back-projection error (below) is minimized***.

$$\sum_i (x_i' - \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}})^2 + (y_i' - \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}})^2$$

Homography has multiple benefits in image processing. This projective transformation can map *any four points* in the plane to any other four. *Rotations, translations, scalings, and shears* are all special cases. Hence, homography can become accustomed to changing the plane and the perspective of an image. Here **Homogeneous Coordinates** are helpful, which increase the dimension of current coordinates. In this case, the **2D points** of the image will become **3D points** in homogeneous coordinates.

Now since a homography is a **3×3 matrix**, it can be written as:

$$H = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix}$$

For mapping, consider set of points - (x_1, y_1) in the first image, and (x_2, y_2) in the second one. The Homography matrix **H** maps in the following way:

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = H \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

Although there are nine entries in the matrix , the homography only contains **eight degrees of freedom**; the entries are redundant concerning scale. In order to estimate the matrix, **4** points in the input image are selected and mapped to the desired locations in the unknown output image according to the use case (*8 equations and 8 unknowns*, which can be easily solved). Once the transformation matrix (**M**) is calculated, the perspective transformation (***cv2.warpPerspective()*** function) is applied to the entire input image to get the final transformed image.

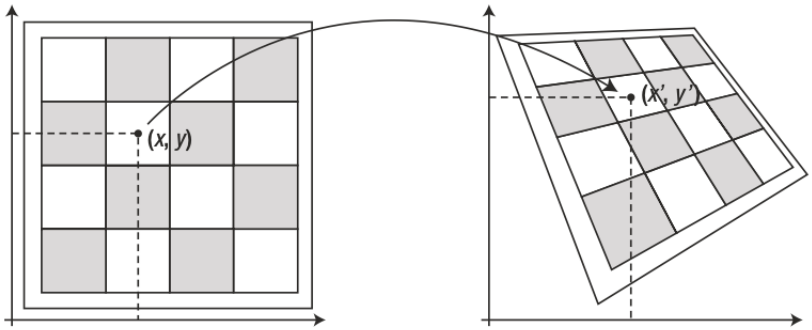

Figure 2. *Homography application to change the perspective of an image*

4 Implementation pipeline

The days were shifted because of the deadline shift.

- Reviewing the literature on the topic and searching for datasets (*April 7*)
- Outlining the algorithm implementation from the LA background (*April 14*)
- Implicitly coding the algorithm (*April 21*)
- Testing the algorithm on datasets, and improving it (if needed) (*April 28*)
- **Second interim report (*May 13*) (We are here :))**
- Implementing image blending algorithm (optional) (*May 24*)
- Preparing for the presentation of the project (*May 26*)
- Presenting the final version of the project and submitting the final report(*May 29*)

5 Literature

1. Alex Caparas, Arnel Fajardo, Ruji Medina, "Feature-based Automatic Image Stitching Using SIFT, KNN and RANSAC", International Journal of Advanced Trends in Computer Science and Engineering, Volume 9, No.1.1, 2020.

2. Matthew Brown and David G. Lowe, "Automatic Panoramic Image Stitching using Invariant Features", Department of Computer Science, University of British Columbia, Vancouver, Canada.

3. Szeliski, R. (2022) Computer Vision: Algorithms and Applications, Second Edition, pp. 511-512.

References

[1] https://en.wikipedia.org/wiki/Harris_corner_detector

[2] Daliyah S. Aljutaili, Redna A. Almutlaq, Suha A. Alharbi, Dina M. Ibrahim , "A Speeded up Robust Scale-Invariant Feature Transform Currency Recognition Algorithm", World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering Vol:12, No:6, 2018.

[3] Chen, S. E. (1995). QuickTime VR – an image-based approach to virtual environment navigation. In ACM SIGGRAPH Conference Proceedings, pp. 29–38.

[4] Szeliski, R. and Shum, H.-Y. (1997). Creating full view panoramic image mosaics and environment maps. In ACM SIGGRAPH Conference Proceedings, pp. 251–258.

[5] https://docs.opencv.org/4.x/da/df5/tutorial_py_sift_intro.html

[6] https://stackoverflow.com/questions/51197091/how-does-the-lowes-ratio-test-work

Keep strong, stay healthy and learn LA!