1 Introduction

The process of focusing on photography is essential for capturing sharp images. Traditional cameras require the user to focus on the subject before taking a photo, and once the photo is taken, the focus cannot be changed. However, cameras with post-capture refocusing capability allow users to adjust the focus on different objects after taking the photo. This is achieved using the focal-stack refocusing approach, which involves two steps. First, a series of photos are taken at different focal lengths by moving the lens. Then, the user selects the sharpest image from this series, depending on the object they want in focus. While this technique emulates the refocusing effect of light field cameras, it requires extra memory to store multiple images.
A key challenge with this method is that as the lens moves to capture different focal planes, objects not on the focal plane appear blurred. Additionally, the changing distance of the lens to the image sensor causes radial image distortions, leading to a less pleasant viewing experience when switching focus between objects. This project aims to find a way to eliminate these distortions, enhancing the user's experience with focal-stack refocusing.

2 Methodology

Our technique is composed of two distinct phases:

1. image alignment
2. sharpness-based image selection

The concept of image alignment involves two fundamental steps: feature detection and matching, and the homography transformation. This process ensures that all images are synchronized in terms of perspective and orientation, creating a cohesive and uniform series of images. The subsequent phase involves selecting the sharpest image from this aligned focal stack thus providing seamless switching.

2.1 Image Alignment

Image alignment is a critical step in the processing of focal stacks, where a series of images captured at different focal depths are seamlessly combined to enhance overall focus throughout a scene. The primary objective of image alignment is to register these images accurately, ensuring that corresponding features across different frames are geometrically matched.

2.1.1 Feature Detection and Matching

Feature Detection and Matching begins with identifying unique, invariant features in each image using an algorithm, Scale-Invariant Feature Transform (SIFT). SIFT detects key points and creates detailed descriptors that encapsulate the local appearance around each point. These descriptors are crucial for the next stage, feature matching. In this stage, we employ the Brute-Force Matcher (BFMatcher) from OpenCV. This matcher method works by comparing feature descriptors from different images. Specifically, it implements a k-nearest neighbors approach, typically with k set to 2. This means for each feature point descriptor in one image, the matcher finds the two closest descriptors in the other image. These pairs are potential matches. To ensure the quality of the matches, we apply Lowe's ratio test. This test compares the distance of the closest match to that of the second closest. If the ratio

of these distances is below a predetermined threshold (commonly 0.7), the match is deemed good or reliable. This ratio test effectively filters out weaker matches, ensuring that only the most robust and distinctive matches are used for the subsequent steps (homography transformation).

2.1.2 Homography Transformation

To compute the homography matrix **H** using matched feature points, we rely on a set of point correspondences between two images (base image and i-th image in the focal stack). Let's consider that we have a set of n matched point s, where each match consists of a point $(x_i,\ y_i)$ in the first image and a corresponding point $(x_i^{'},\ y_i^{'})$ in the second image. The goal is to find the homography matrix H that relates these points. The homography matrix is a 3×3 matrix given by:

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

The relationship between the points in the two images under the homography transformation is:

$$\begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} = H \cdot \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \tag{2}$$

The process of aligning images within the focal stack involves utilizing the findHomography function provided by OpenCV, which yields a transformation matrix denoted as **H** (homography matrix). This matrix allows for the seamless alignment of all images concerning a chosen base image.

Aligned Image without Cropping.

Aligned Image with cropping.

Figure 1: Boarder Removing.

A common challenge in image alignment arises from the presence of black borders, visible in the left image of Fig 1. These borders emerge as a consequence of the warping process during homography, where parts of the image may shift beyond their original boundaries, leaving empty areas often filled with black pixels. To maintain visual consistency and eliminate unwanted black borders, we employ a cropping strategy. This involves removing the black edges, resulting in the visually cleaner image

depicted on the right in Fig 1. This cropping approach ensures that all images in the focal stack are adjusted to the same dimensions. Consequently, alignment is achieved without any extraneous elements, enhancing the overall quality and cohesion of the image stack.

2.2 Sharpness-Based Image Selection

This method involves applying a Laplacian kernel to highlight edges in each image, using a box filter to calculate a local sharpness measure, and then selecting the image with the maximum sharpness at each pixel. This comprehensive approach ensures the identification of the sharpest focus for each pixel across the aligned focal stack.Given a set of k images $\{I_1, I_2, ..., I_k\}$, each of size $m \times n$, we perform the following operations:

2.2.1 Gradient Calculation with Laplacian Convolution

For each image I_j in the set, we apply we apply a Laplacian convolution to compute its gradient. This operation is denoted as $G_j = I_j * L$ where G_j represents the gradient map of the image I_j and L refers to the Laplacian kernel. A commonly used form of the Laplacian kernel is a 3x3 matrix shown below.

$$L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

2.2.2 Sharpness Measurement with Box Filter

Each gradient map G_j is then processed with a box filter convolution to calculate the sharpness measure, represented as $S_j = G_j * B$.In this equation, S_j is the sharpness measure of the image I_j, and B is the box filter.

2.2.3 Selecting the Sharpest Image in the focal stack at a Given Point

The algorithm then focuses on a specific point (x, y) across all images. For each sharpness measureS_j, the value at this point is extracted. The image with the highest sharpness value at (x, y) is selected. This is done by finding the index j_{max} which corresponds to the image with the maximum sharpness at the specified point satisfies $j_{max} = \arg\max\{S_1(x, y), S_2(x, y), \ldots, S_k(x, y)\}$.

3 Implementation

3.1 Focal Stack preparing

In our experiment, we utilized the Canon EOS 7D, known for its advanced autofocus system with 19 cross-type focus points, to capture images. A key aspect of this AF system is the "Single Point AF" mode, allowing for manual selection of any one of the 19 focus points shown in Fig 2. This feature is particularly advantageous for precise focusing, especially in compositions where the subject is off-center or in challenging focus scenarios. Utilizing this mode, we took 19 photos of a static scene to create a focal stack. We strategically placed objects in various parts of the scene - foreground, middle, background, and both sides - to facilitate detailed observation.

Figure 2: Single Point AF mode.

3.2 Results

We implemented our method on a laptop. By clicking the cursor on the image window, based on the pixel clicked, our program would pick an appropriate image from the focal stack where that pixel is in the sharpest focus. In Figure 3a, two images from the original focal stack are overlapped to create the visual effect, while Figure 3b shows the result of aligning and perfectly matching two rectified images from the same stack. This contrast highlights the enhanced alignment and matching accuracy in Figure 3b compared to the overlapped presentation in Figure 3a.

(a)Before alignment.

(b)After alignment.

Figure 3: Two Images in the focal stack are overlapped.

We also implement the all-in-focus feature, we applies a Gaussian kernel to each image and then computes the value of the Laplacian gradient which is G_j. This process helps quantify the edge strength in the image, identifying focus regions. The program then creates a new image and, for each pixel, selects the pixel with the strongest edge (highest gradient value) from the stack of images. This selected pixel, with its RGB values, is copied into the output image. As a result, each pixel in the final image is the sharpest version of itself from the entire stack, leading to an image where all regions are in focus show in Fig 4.

Figure 4: All In Focus.

References

[1] *Depth-Focal-Stack*, Available at: `https://github.com/hosseinjavidnia/Depth-Focal-Stack`.

[2] *Fast and easy focus stacking*, Available at: `https://github.com/PetteriAimonen/focus-stack/tree/master/examples`.

[3] *Focus-Stacking*, Available at: `https://github.com/momonala/focus-stack/blob/master`.

[4] *Application of Preconditioned Alternating Direction Method of Multipliers in Depth from Focal Stack*, Available at: `https://github.com/hosseinjavidnia/Depth-Focal-Stack/tree/master`.

[5] *Image Alignment (Feature Based) using OpenCV (C++/Python)*, Available at: `https://learnopencv.com/image-alignment-feature-based-using-opencv-c-python/`.

[6] *Image Alignment (ECC) in OpenCV (C++/Python)*, Available at: `https://learnopencv.com/image-alignment-ecc-in-opencv-c-python/`.

[7] *Homography (computer vision)*, Available at: `https://en.wikipedia.org/wiki/Homography_(computer_vision)`.