Implementation of the SIFT method

Prepared by
Aimee Valladares

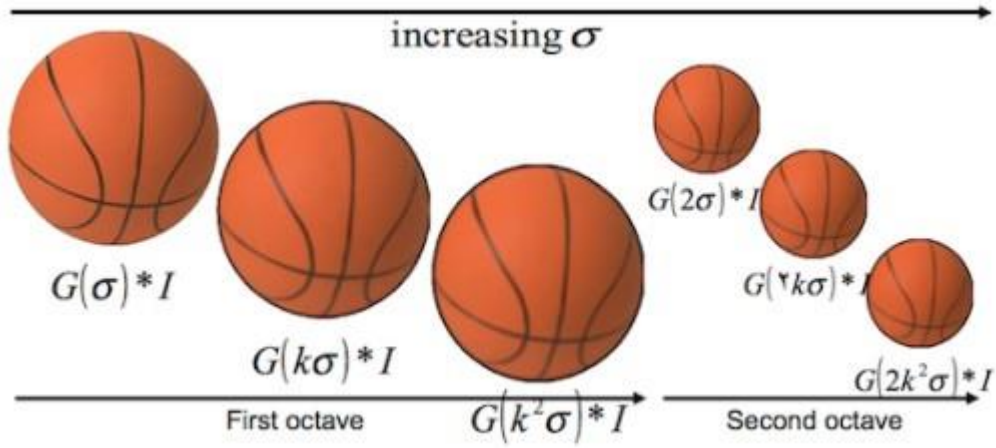
Computer Vision
CAP 4410

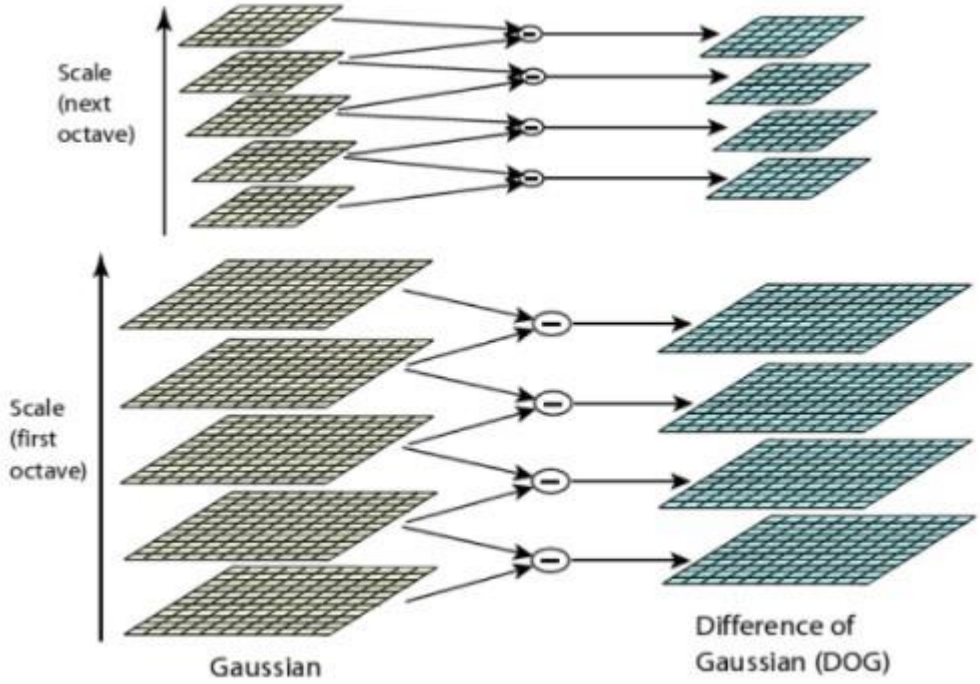
4 November 2020

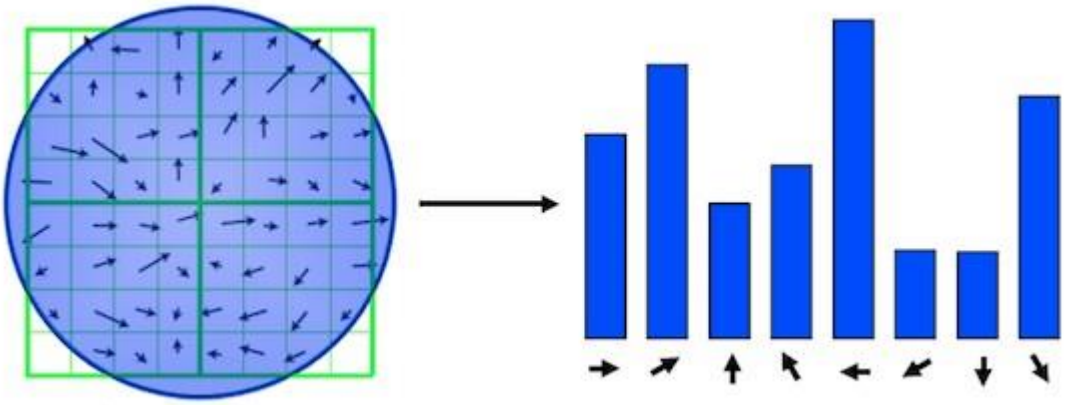
Florida Polytechnic University

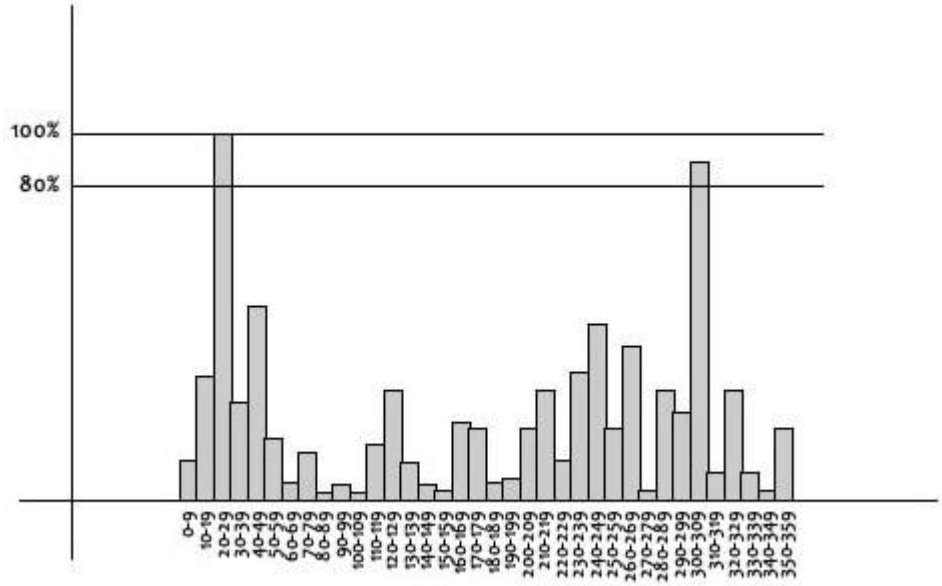
Table of Contents

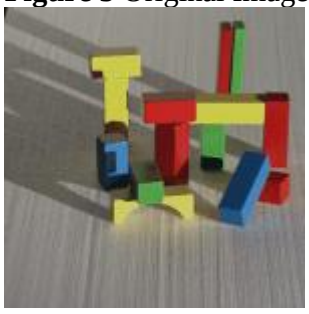
List of Figures

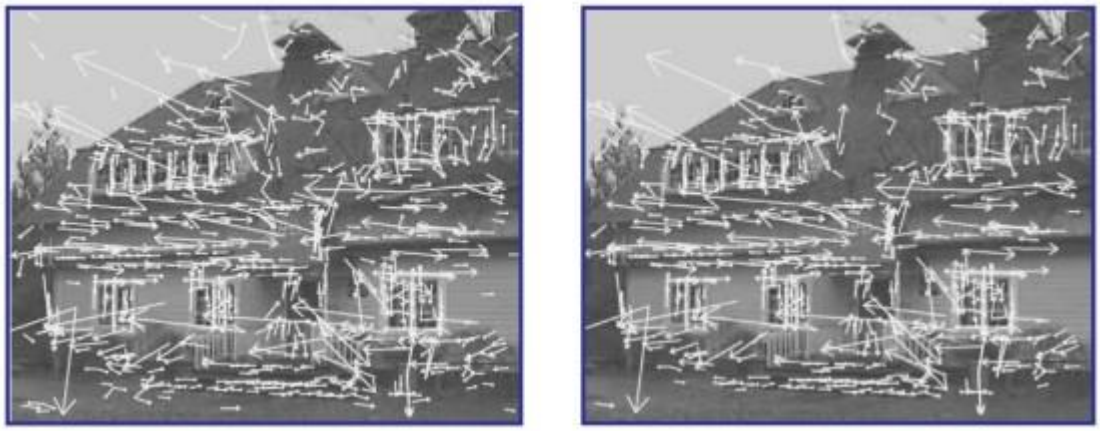


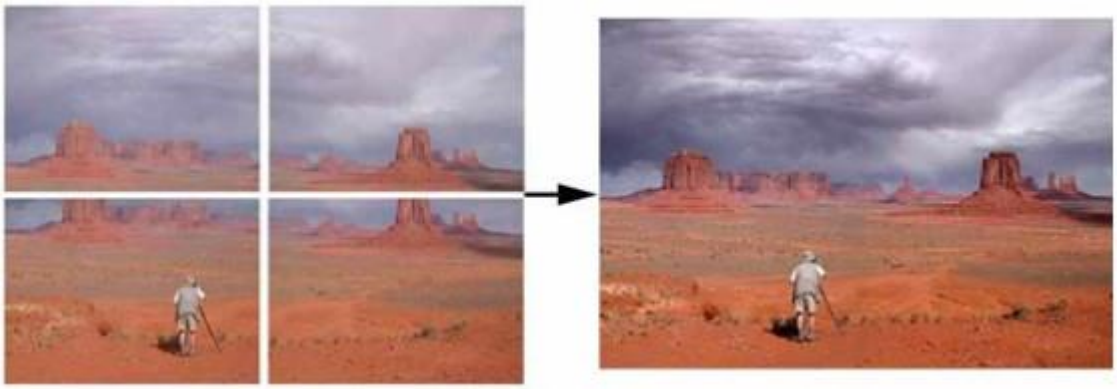


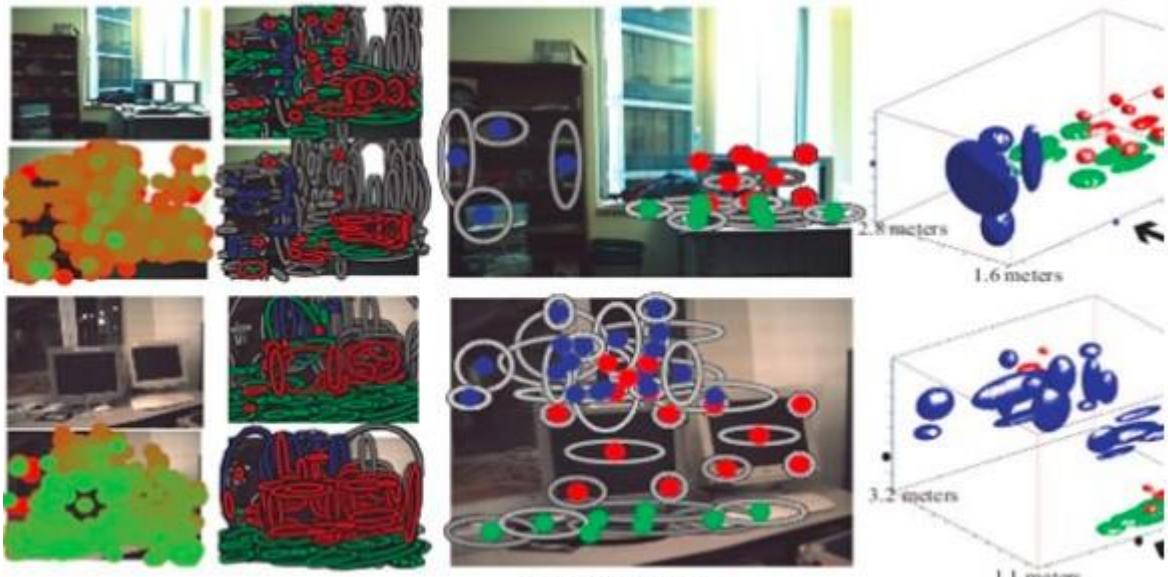












Stereo Test Images

List of Equations

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

* is the convolution operator

$G(x, y, \sigma)$ is a variable-scale Gaussian

$I(x, y)$ is the input image

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)$$

$L(x, y, k\sigma)$ is a variable-scale Laplacian of Gaussian

$L(x, y, \sigma)$ is a Laplacian of Gaussian

$$D(\mathbf{x}) = D + \frac{\partial D}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}$$

$$\mathbf{x} = (x, y, \sigma)^T$$

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1} \frac{\partial D}{\partial \mathbf{x}}$$

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \qquad Tr(H) = D_{xx} + D_{yy} = \lambda_1 + \lambda_2$$

$$Det(H) = D_{xx}D_{yy} - (D_{xy})^2 = \lambda_1 \lambda_2$$

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

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

1. The SIFT Method

1.1 Introduction

Matching different images based on certain features is a difficult task due to certain factors not being similar presenting an issue known as the *Scale Invariant Problem*. For instance, if the images have different scales or rotation then corner edge detectors like the Harris Corner Detector would not accurately match the features of the images. As a result, different methods can be utilized to complete this difficult task. These possible methods include edge detectors, the SURF (Speeded Up Robust Features) method, the SIFT method, etc. The approach we will be focusing on today is implementing the scale-invariant feature transform (SIFT) method to resolve the problem. The SIFT method is a feature detection algorithm that detects and describes the local features in images [1].

The SIFT method is a common method utilized for matching different images based on certain features due to its many advantages. One advantage this method is that it focuses on the local features which are robust to occlusion and clutter. Another advantage is that the method is very distinctive as the individual features can be matched to a large database of objects. Another advantage is that the method can generate countless of features for objects of varying size either small or large. Finally, another advantage the method had was that it was very efficient due to it being close to real-time performance [2].

1.2 Project Description

The SIFT method transforms the image data into scale-invariant coordinates and extracts the distinctive invariant features. The SIFT method tries to complete this transformation by going through key stages. The SIFT method typically consists of four key stages: Scale-space Extrema Detection, Key Point Localization, Orientation Assignment, Key Point Descriptor, and Key Point Matching. In this project we are focusing on how certain key stages for the SIFT method would affect the given image. The key stages that would be demonstrated and focused on the program are scale-space extrema detection, key point localization, and orientation assignment.

The scale-space extrema detection stage tries to detect the locations in the image that are identifiable from different scales of the same object. This process is done by trying to ding the Difference of Gaussians to locate the scale-space extrema. The key point localization stage tries to eliminate the key points generated by the scale-space extrema detection stage by finding the key points that either have low contrast or are poorly localized on an edge. The orientation assignment stage then focuses to assign a consistent orientation to the key points based on the local image properties by quantifying the local image characteristics [3].

The main objective of this project is to utilize and demonstrate how each key stage of the SIFT method affects the image. This implementation will be done by creating an OpenCV project that will apply each key stage of the SIFT method. These programs will demonstrate the effects of how the each of the key SIFT stages help locate the local features in the given image.

1.3 Program Design

The OpenCV project in Visual Studio will comprise several C++ files that will implement and demonstrate each key stage of the SIFT method. The OpenCV code will comprise of three C++ files with each file demonstrating the first three key stages of the SIFT method.

The first program is meant to illustrate the first key stage of the SIFT method, the scale-space extrema detection. The program constructs a scale space that will help figure out the scale invariant features in the given image. This is done by increasing the octave in several scale levels, for this program we will do four octaves. An example of this step is seen in Figure 1. The program will then convolve the image with the Gaussian filters, specifically with the Laplacian of Gaussian, at different scales. The program will then find the Difference of Gaussian by finding the difference of the consecutive Gaussian-blurred images at different scales. These processes are demonstrated in Figure 2, Equation 1, and Equation 2. This technique is used to detect the stable key points in the scale space. Finally, the results will be the local maxima and minima of the Difference of Gaussians [3] [4] [5].

The second program is meant to illustrate the second key stage of the SIFT method, the key localization. This program will build upon the previous program as the scale-space extrema detection stage generates too many key points. This program is meant to eliminate the key points that are unstable specifically the key points that have low contrast or are poorly localized on an edge. When trying to find the low contrast candidates, the program calculates the Taylor series expansion of the Difference of Gaussian, as seen in Equation 3. This calculation is simplified in the program through the use of the **contrastThreshold** function. This function removes the detected key points that has an intensity lower than the specified threshold. When trying to eliminate key points relating to the edge the program will go through a process like the Harris corner detector. However, the program will also compute the Hessian matrix of the determinant for the principal curvature, as seen in Equation 4. Finally, the program will remove the outliers by evaluating the ratio of the calculated eigenvalues, seen in Equation 5. These calculations are simplified in the program through the use of the **edgeThreshold** [2] [4] [6].

The third and final program is meant to illustrate the third key stage of the SIFT method, the orientation assignment. The program will build upon the previous program as the key localization results will establish valid key points. This program will try to assign a consistent orientation to the key points based on the given image. This is done to realize invariance to the image rotation. The program first computes the gradient magnitudes and orientations at the scale of the key point, which can be seen in Equation 6 and Equation 7. The program then creates a histogram of the resulting key points. The histogram is then broken into 36 bins (which is meant to cover the 360 degrees of orientation) and is weighted by the gradient magnitudes and the spatial gaussian filter with the scale of the key point. Finally, the program selects the peak of the histograms as direction of the key point. Then, if any of the local histogram peaks are above 80% of the highest peak are converted into new key points with the same location and scale, but different directions. An example of these processes can be as seen in Figure 3 and Figure 4 [2] [4] [7].

2. The OpenCV Project

2.1 Program Code

The OpenCV Project was written in C++ with Visual Studio with the program consisting of three files. The program source code is included in the ZIP folder provided. The files are well-commented, with extended comments used to help distinguish between the three files and help explain the logic of the code [8] [9] [10] [11].

2.2 Test Plan

To properly demonstrate the performance and accuracy of the OpenCV project code, we run each of the programs and analyze the outputs. To analyze whether the programs accurately demonstrate each of the key SIFT stages we would compare the resulting image to a sample image that established that key stage. For instance, if the program meant to simulate the first key stage and that program generates an image, then that image should be similar to the provided sample seen in the "List of Figures" section of this report.

Typically, each of the programs will generate several images in separated window. The first generate window will display the original unedited image. The second generated window will display the original image illustrate a key SIFT stage.

The first program will generate three images: the original image, the octave image, and the resulting key point image. Due to the program implementing the scale-space extrema detection key SIFT stage, the modified images will essentially be gray-level images. The octave image will progressively blur out the original image. Meanwhile, the resulting key point image will have the original gray-level image along with countless highlighted key points.

The second program will generate three images: the original image, the scale-space extrema detection image, and the key point localization image. Due to the program implementing the key point localization key SIFT stage, the modified images will essentially be gray-level images. The scale-space extrema detection image should be the same result as the first program. Whereas the key localization image should be a gray-level image with fewer highlighted key points when compared to the scale-space extrema detection image.

The third program will generate three images: the original image, the key localization image, and the orientation assignment image. Due to the program implementing the orientation assignment stage and building upon the previous two stages, the modified images will essentially be gray-level images. The key localization image should be the same as the second program. Whereas the orientation assignment image should be a gray-level image with key points indicating the orientation and direction.

2.3 Test Results

The OpenCV project should be able to perform correctly though there a margin of error for the accuracy of the implementation of each program, representing certain key SIFT stages. The results of the program do accurately reflect how each of the key stages affects the given image. This is illustrated by the program having resulting images that are similar to the images seen between Figures 6 - 9.

Figures 6 and 7 are the control images that depicts how the first program results should behave like the first SIFT key stage, scale-space extrema detection. When comparing the program results to the figures seen in the "List of Figures" section of this report, the images should be similar. The program should illustrate the octave images and the initial key points generated by the scale-space extrema detection key stage.

Figures 7 and 8 are the control images that depicts how the second program results should behave like the second SIFT key stage, key localization. When comparing the program results to the figures seen in the "List of Figures" section of this report, the images should be similar. The program should illustrate the initial key points generated by the scale-space extrema detection key stage and the new key points generated by the key localization. Specifically, the results should illustrate a difference in the amount of key points as the second program should reduce the amount of key points by eliminating outliers that have low contrast and are poorly localized to the edge.

Figures 8 and 9 are the control images that depicts how the third program results should behave like the third SIFT key stage, orientation assignment. When comparing the program results to the figures seen in the "List of Figures" section of this report, the images should be similar. The program should illustrate the difference in key points from the results of the second program. Specifically, the program should illustrate how finding out how the orientation aspect to the image affects the accuracy of the generated key points for the local features of the image.

2.4 Conclusion

The SIFT method is a common method to use when trying to match different images. As a result, this method has many applications including object recognition and tracking, image stitching, 3D modeling, and robot mapping. Examples of these applications can be between Figures 10 and 11. In the project, we learned about certain key stages involved in the SIFT method. Specifically, we learned how each stage affects the resulting local features in a given image as well as learned how the common patented SIFT method goes about solving the Scale Invariant Problem when trying to match different images with the same local features.

In the future, I would like to make several improvements that would help improve the accuracy of my programs. One improvement I would make to the OpenCV project was to go into detail about the fourth and final key stage, key descriptor, if given more time. Another improvement I would make to the OpenCV project was to include more octaves as doing so would possibly give more accurate resulting key points. Another improvement I would make to the project was to compare how possibly applying a different smoothing filter to the scale-space extrema detection stage to see if that would affect what initial key points would be generated. Especially, when taking into account

the process of determining the octaves of the image. Finally, one improvement I would make to the project is to try to augment the local invariant features of the SIFT method by calculating the global features of an image to see if this change can generate fewer areas with ambiguity when trying to match local descriptors.

References

[1] "Scale-Invariant Feature Transform." *Wikipedia*, Wikimedia Foundation, 20 Oct. 2020, en.wikipedia.org/wiki/Scale-invariant_feature_transform.

[2] Abid, Muhammad, Lecture 12 Slides, CANVAS *Concise Computer Vision*

[3] *SIFT Image Features*, homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV0405/MURRAY/SIFT.html.

[4] TZstatsADS. "TZstatsADS/ADS_Teaching." *GitHub*, github.com/TZstatsADS/ADS_Teaching/blob/master/1-Spring2016/Tutorials/wk7-image_analysis/advanced_image_analysis.md.

[5] Sinha, Utkarsh. "LoG Approximations." *AI Shack*, aishack.in/tutorials/sift-scale-invariant-feature-transform-log-approximation/.

[6] "Introduction to SIFT (Scale-Invariant Feature Transform)." *OpenCV*, docs.opencv.org/3.4/da/df5/tutorial_py_sift_intro.html.

[7] Sinha, Utkarsh. "Keypoint Orientations." *AI Shack*, aishack.in/tutorials/sift-scale-invariant-feature-transform-keypoint-orientation/.

[8] Vanero. "Vanero/SIFT-1." GitHub, github.com/vanero/SIFT-1/tree/master/MySIFT.

[9] Tqjustc, et al. "How to Use SIFT in Opencv." *Stack Overflow*, 1 June 1963, stackoverflow.com/questions/22722772/how-to-use-sift-in-opencv.

[10] Computer Vision Official. "Step by Step Guide to Extract SIFT Features from Image Using OpenCV C++." *Youtube*, 1 Oct. 2019, www.youtube.com/watch?v=mHWjdav8KVk.

[11] "Introduction to SIFT (Scale-Invariant Feature Transform)." *极客分享首页*, www.geek-share.com/detail/2719833489.html.

[12] Chopra, Eklavya. "Feature Detection as in 1999: SIFT Explained with Python Implementation." *OpenGenus IQ: Learn Computer Science*, OpenGenus IQ: Learn Computer Science, 6 Sept. 2019, iq.opengenus.org/scale-invariant-feature-transform/.

[13] Choudhury, Rahul. *Recognizing Pictures at an Exhibition Using SIFT*. Standford University, web.stanford.edu/class/ee368/Project_07/reports/ee368group11.pdf.

[14] "Distinctive Image Features from Scale-Invariant Keypoints By David G. Lowe, University of British Columbia Presented by: Tim Havinga, Joël Van Neerbos. - Ppt Download." *SlidePlayer*, slideplayer.com/slide/6276090/.

[15] Cardoso, Samantha, et al. *Understanding SIFT Algorithm and Its Uses*. International Journal of Science Technology & Engineering, Apr. 2016, www.ijste.org/articles/IJSTEV2I10141.pdf.

Volume 2. Issue 10.