Proceedings of 13th IOE Graduate Conference
Peer Reviewed
ISSN: 2350-8914 (Online), 2350-8906 (Print)
Year: 2023 Month: April Volume: 13

Application of Computer Vision in Robotic Arm

Sushan Kattel [a], Sudesh Mate [b], Kushal Bhattarai [c], Binilraj Adhikari [d], Manoj K. Guragai [e]

[a, b, c, d, e] *Purwanchal Campus, IOE, Tribhuvan University, Nepal*

[a] sushan074bex@ioepc.edu.np, [b] sudesh74bex@ioepc.edu.np, [c] kushal074bex@ioepc.edu.np, [d] binilraj74bex@ioepc.edu.np, [e] mkguragai@gmail.com

Abstract

Talking about large scale industrial works, humans are not capable of efficiently performing long shifts of repetitive labor. The robotic arm featured in this paper, i.e., APPLICATION OF COMPUTER VISION IN ROBOTIC ARM is capable of picking and dropping objects from disorganized locations to an organized defined location. The entire system is controlled by Raspberry pi 4. Image files were taken as input which was processed by converting firstly into Hue, Saturation and Value (HSV) format. Other various computer vision techniques like Histogram Equalization, Circular Hough Transform (CHT) and Canny Edge Detection were used to locate the center point of the circular objects. From the findings, color processing performs best in circumstances with consistent lighting, but it may not be accurate when there is a complicated background or dynamic lighting. In these situations, using CHT to follow the target object should be the main focus of the object detection strategy. The use of linear regression using data gathered from several trials has been shown to be an effective method for determining constant values with little error. Similarly for picking the target objects from the locations, the authors employed inverse kinematics on a 4-Degree of Freedom (DOF) system to set the end effector to the targeted location. All these techniques were combined to operate the computer vision controlled robotic arm which successfully detected and collected the required red colored object in a container and discarded the blue colored one. This system is highly precise, accurate and reduces human intervention in risky industrial areas and is useful in agriculture.

Keywords

Histogram equalization, Canny edge detection, Inverse Kinematics, Color Processing, Linear Regression

1. Introduction

Robotic Automation is favored since humans are not always efficient to perform a variety of tasks such as lifting heavy loads, working for continuous long hours, or performing high precision work. In this work, a system that can detect red and green colored circular objects is developed and implemented using a 4- DOF robotic arm for pick and place operation. Object detection and tracking is an important area of computer vision research and development that uses a sequence of frames to identify, recognize, and track things. Using appearance-based or feature-based approaches, object identification algorithms rely on matching, learning, or even pattern recognition techniques.

In terms of object detection and tracking, robotic automation is necessary to reduce the operator's need to be physically present on site to do the tasks assigned to him. Detecting objects against a complicated backdrop is a difficult task in image processing. The purpose of this work is to use color processing and the CHT to find red and green colored objects in a complicated backdrop image.

The Python programming language is utilized to fulfill the research's aim. It includes OpenCV, a cutting-edge image processing and computer vision library that is used for image processing. Although color processing has the best probability of detecting red and green objects, it can only detect the targeted item in static lighting circumstances. The colors of the targeted item may shift nonlinearly depending on the lighting circumstances. The color will vary greatly based on the strength of the light, the item's reflected rate, and the object's background, which may or may not produce the same Red Green Blue (RGB) color as the object. Because this work may involve detecting inconstant lighting conditions and a complicated background, the object detection approach may need to concentrate on utilizing CHT. In order to autonomously position the robotic arm for pick and place operations, the system uses inverse kinematic algorithms. The arm used for this research is MK2 [1]. This arm prototype is meant to demonstrate the proper functionality and scalability of the system, not to include every desired system feature. There are countless features and devices that the system could eventually incorporate, but in order to maintain proper research scope, only 4-DOF is used.

2. Previous Works

There has been research into how computer vision techniques can be used to control the robotic arm's mobility. To detect sunflower and mango, S. Bindu et al (2014) employed the Circular Hough Transform [2]. The highest success rate according to this research was 96.2 % . In detection of the mangoes, 4 mangoes out of 5 were detected and in the detection of the sunflowers, 2 sunflowers out of 3 were detected. Similarly in the detection of the buttons, 24 buttons were detected out of 25. To recognize the mango from a mango tree, R. Hussain et al (2012) employed edge detection and CHT [3]. It successfully detected the object from the background image. L. Fernandes et al. (2020) employed a median filter to remove noise, HSV range-based color identification, and contour detection approach to detect from, as well as a robotic arm to sort the discovered item [4]. Three colors red, blue and green were used and 3 primary shapes triangle, square and rectangle were used. The designed system had successfully identified all the considered

shapes and colors of the objects. M. Intisar et al. (2020) employed contouring to identify the objects [5]. By applying this technique, they have identified objects based on color, shape, and size without using multiple techniques for each parameter. Similarly, they created an interactive and simple Graphical User Interface (GUI) for ease of operation and high degree of control over the operation of the system. This system also successfully determined the object, picked it up and placed it on the desired location. Similarly, W. G. Hao et al. (2011) employed Universal Synchronous/Asynchronous Receiver/Transmitter (USART) communication protocol to communicate between software and robotic arm hardware for efficient functioning of the robotic arm [6]. They firstly designed a virtual arm in DirectX and then created an actual robotic arm which performed simple pick and place operations in a practical environment.

3. Methodology

The recognition process that is used in this work can be shown in the form of a block diagram. Python3 [7] is used as programming language for this work and image processing and manipulation is done using OpenCV library [8].

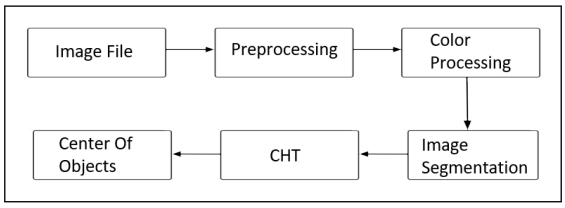

Figure 1: Block diagram of the Object recognition process

The image is first passed via the preprocessing block, where histogram equalization was used to improve the image. Histogram equalization tends to boost the image's contrast, resulting in a better outcome for region-based feature extraction. The irrelevant color and unrelated object must then be removed from the original image during color processing. The threshold procedure can be used to eliminate unrelated objects.

The picture segmentation must now be completed. To do segmentation, edge detection must be used. The edge detection mechanism is critical since the CHT approach requires edge information. In this work, canny edge detection is employed among numerous edge detection approaches. After the detection of a circular red colored object, now the task is to localize the object in the frame and use inverse kinematics to map the coordinates of the target with the angular position of the servo motors guiding the motion of the robotic arm.

To pick the object by calculating degrees of servo motor from given coordinates the following algorithm is used:

1. Start.
2. Turn on the system.
3. Initialize position of servo motors.
4. Capture the real time video frame by frame.
5. Detect and localize the red colored circular object.
6. Calculate the angle required to reach the target using inverse kinematics.
7. Grab the target using the end effector and place it in the required container.
8. Jump to step 3.

The step 5 is broken down to following four parts:

- Histogram Equalization
  The histogram is a graphical depiction of an image's intensity distribution. It simply indicates the number of pixels for each intensity value taken into account. Histogram Equalization is a computer-aided image processing technique for enhancing picture contrast. It does so by effectively spreading out the most common intensity values, i.e., expanding out the image's intensity range. When the useful data is represented by near contrast values, this approach frequently boosts the global contrast of pictures. This enables locations with poor local contrast to obtain a boost in contrast.
- Threshold Operation
  Color Threshold is a way for removing sections of a picture that fall inside a specific color range. This technique may be used to find items that have the same color values.
- Canny Edge Detection
  The Canny edge detector is an edge detection operator that detects a wide range of edges in pictures using a multi-stage approach. The Canny edge detection technique may be broken down into five distinct steps:
  - Smooth the image with a Gaussian filter to eliminate the noise.
  - Locate the image's intensity gradients.
  - To eliminate erroneous edge detection responses, use gradient magnitude thresholding or lower bound cut-off suppression.
  - Determine probable edges using a twofold threshold.
  - Hysteresis edge tracking: Finish edge detection by suppressing all edges that are weak and not coupled to strong edges.
- Circular Hough Transform
  The Circle Hough Transform (CHT) is a fundamental feature extraction approach for recognizing circles in defective pictures in digital image processing. The circle candidates are created by picking local maxima in an accumulator matrix and then "voting" in the Hough parameter space.

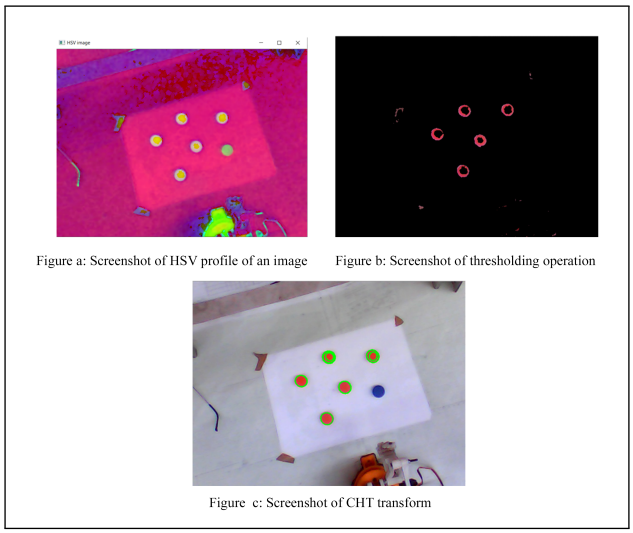

Figure 2: Image Transformations

3.1 Inverse Kinematics

Kinematics is the study of motion without considering the cause of the motion, such as forces and torques. Inverse kinematics is the use of kinematic equations to determine the motion of a robot to reach a desired position. In this research, the authors take a picture from a camera and thus detect the pixel coordinates and convert into desired coordinate value. Thus, after having the given coordinates, it is required to find the angles for the motors so that the end effector can reach the desired location. Numpy [9] is used for these calculations. The author's configuration is given in figure 3.

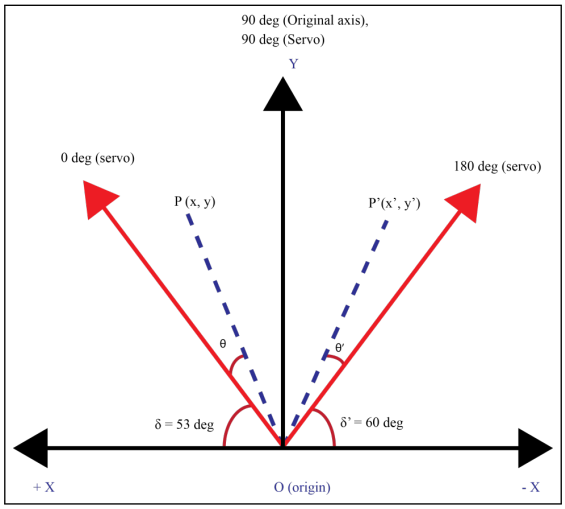

Figure 3: Base configuration of our arm

For the calculation of base angle, the authors have studied figure 3. For the left half plane,

$$\phi = \delta + \theta \tag{1}$$

$$\phi = \tan^{-1}(\frac{y}{x}) \times \frac{180}{\pi} \tag{2}$$

$$\theta = \phi - \delta = \phi - 53 \tag{3}$$

Similarly, for the right half plane

$$\phi' = \delta' + \theta' \tag{4}$$

$$\phi' = \tan^{-1}(\frac{y'}{x'}) \times \frac{180}{\pi} \tag{5}$$

$$\theta' = \phi' - \delta' = \phi' - 60 \tag{6}$$

The general configuration of robotic arm is shown in figure 4.

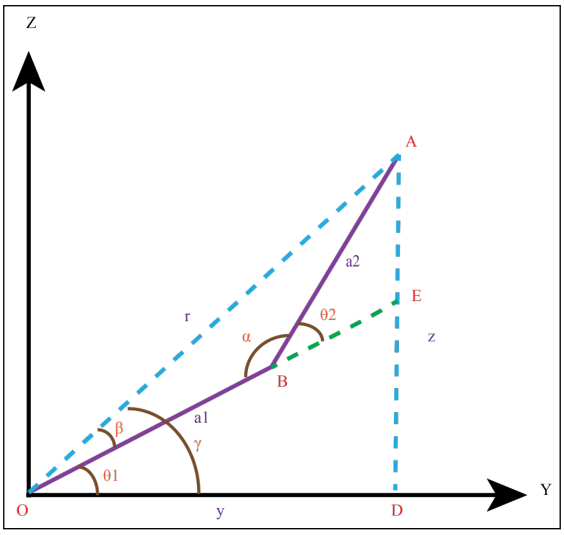

Figure 4: General configuration of robotic arm

From the figure above the authors calculated the following equations for the arm:

$$\theta_2 = \cos^{-1}\left(\frac{y^2 + z^2 - a_1^2 - a_2^2}{2a_1a_2}\right) \tag{7}$$

$$\theta_1 = \gamma - \beta = \tan^{-1}\left(\frac{z}{y} - \tan^{-1}\left(\frac{a_2 \sin\theta_2}{a_1 + a_2\cos\theta_2}\right)\right) \tag{8}$$

Similarly, the figure 5 demonstrates this work's version of robotic arm:

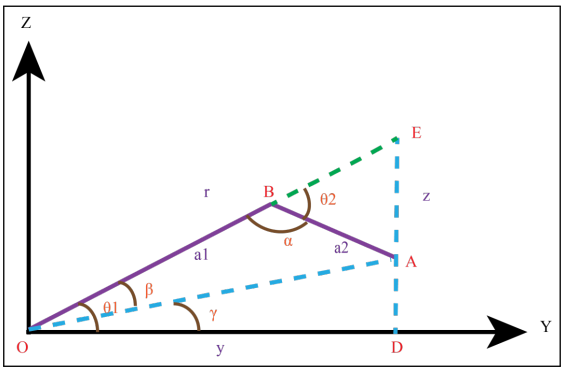

Figure 5: This work's configuration of robotic arm

For this work's configuration the equations (7) and (8) can be modified as shown below:

$$\theta_2 = \cos^{-1}\left(\frac{y^2 + z^2 - a_1^2 - a_2^2}{2a_1a_2}\right) \tag{9}$$

$$\theta_1 = \gamma + \beta = \tan^{-1}\left(\frac{z}{y}\right) + \tan^{-1}\left(\frac{a_2 \sin\theta_2}{a_1 + a_2\cos\theta_2}\right) \tag{10}$$

Applying the following values from this work's system to equation (1) and (2), we get:

$$z = 1.5(constant)$$

$$a1 = 13.5cm$$

$$a2 = 14.7cm$$

$$\theta_2 = \cos^{-1}\left(0.00252y^2 - 1\right) \tag{11}$$

$$\theta_1 = \tan^{-1}\left(\frac{1.5}{y}\right) + \tan^{-1}\left(\frac{14.5\sin\theta_2}{13.5 + 14.7\cos\theta_2}\right) \tag{12}$$

3.2 Linear Regression to Map Values

Linear regression analysis is used to predict the value of a variable based on the value of another variable. They describe relationship between variables by fitting a line to the observed data. In this work's system, linear regression is used to map the physical angles of arm to the angles of the servo motor. Linear regression is calculated using SciPy [10]. For this the authors have calculated slope and intercept values for angles θ , θ' , θ_1 & θ_2 which are given as follows:

$$For\theta \ : Slope = 1.104138687376643$$
$$Intercept = -7.907639138258979$$
$$For\theta' \ : Slope = -1.064553956960368$$
$$Intercept = 185.59185678906084$$
$$For\theta_1 \ : Slope = -0.8308855674128404$$
$$Intercept = 196.9194931259962$$
$$For\theta_2 \ : Slope = -0.5588039245426362$$
$$Intercept = 172.17698010728947$$

Now since the workspace is not uniform, the authors have to map the real angle and servo angle which are shown in figure 3 for which the values for positive half are:

$$Real37 = Servo90$$
$$Real1 = Servo2.43$$
$$\therefore \text{ServoAngle} = \theta \ \times \ 2.43$$

$$\text{Base angle} = \text{Slope} \times \text{ServoAngle} + \text{Intercept} \tag{13}$$

Similarly, for negative half the relationship between real angle and servo angle are given as follows:

$$Real30 = Servo90$$
$$Real1 = Servo3$$
$$\therefore \text{ ServoAngle} = \theta' \ \times \ 3$$

$$\text{Base angle} = \text{Slope} \times \text{ServoAngle} + \text{Intercept} \tag{14}$$

Similarly, for θ_1 & θ_2 the servo angles are calculated by using following expressions:

$$\text{ServoAngle}\theta_1 \ = \text{Slope} \times \ \theta_1 \ + \text{Intercept} \tag{15}$$

$$\text{ServoAngle}\theta_2 \ = \text{Slope} \times \ \theta_2 \ + \text{Intercept} \tag{16}$$

3.3 Mapping Pixel Coordinates with Real coordinates

Since our coordinate is firstly taken in pixels, it has to be mapped into the standard Cartesian coordinate system. The following are the scaling factor and scaling matrix to convert the coordinate system. The pixel coordinate has its origin axis different from the axis of arm. To determine the scaling factor, graph paper was taken and it's one unit coordinate was mapped with pixel coordinates appearing on the image. The block diagram showing its conversion process is shown in figure below:

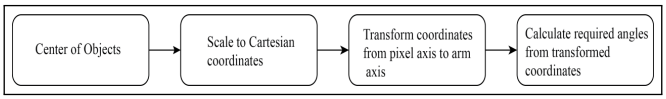

Figure 6: Block diagram showing mapping of coordinates

The following expressions are used to perform the operation shown in figure 6:

$$\text{Scaling factor}(S_x, S_y) = \frac{\text{Dimension of real coordinate}}{\text{Dimension of pixel coordinate}} = \frac{1}{12.45} \tag{17}$$

$$\begin{bmatrix} P_{ix} \\ P_{iy} \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x_{pixel} \\ y_{pixel} \end{bmatrix}$$

Now for the transformation of origin, the authors use the following expression:

$$(arm_x, arm_y) = (P_{ix} - 23.6, P_{iy} + 7.4) \tag{18}$$

Now the required angles are calculated using Inverse kinematics as discussed above in section 3.1.

4. Result and Analysis

The authors have controlled all servo motors using Raspberry Pi 4 [11] via a 16-channel servo controller and thus it can be observed that the system detects the red object and discards the blue colored one. The input feed by the webcam is fed to the image processing algorithms which locate the given object and the arm angles are set such that the end effector will be exactly on the position from where it can pick it up and keep it on the container kept outside workspace. Only the defined color objects are detected and the undefined colors are not taken into consideration by the system.

From the observations the red colored balls were successfully picked and kept in the container and the blue ones were left as it is.

From the data the authors have observed there were some errors that occurred which were corrected on x axis as shown in equations below:

$$\text{For } x > 0 : x = x + 0.5$$
$$\text{For } x < 0 : x = x - 0.5$$

The error factor in y lies in between 1.5 and 3.5, thus on averaging the error on y can be corrected by:

$$y = y + 2.5$$

The following plot shows relationship between real and calculated values. The calculated values are the error corrected values taken from the system. The graph is presented below which is generated using matplotlib [12].

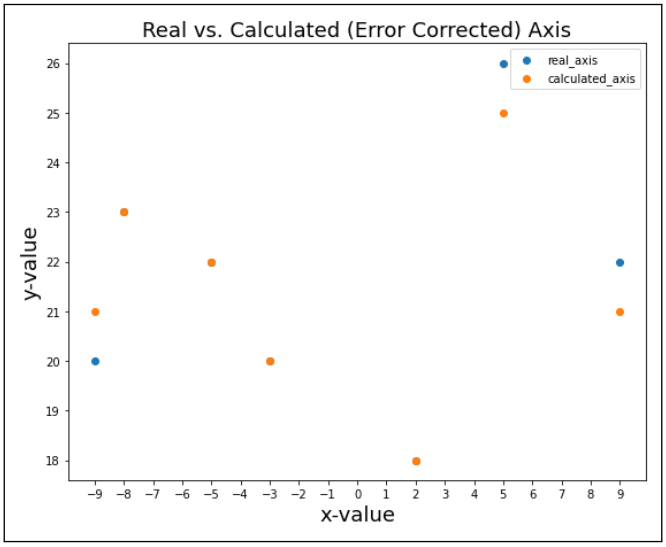

Figure 7: Coordinate values for Real and calculated axis

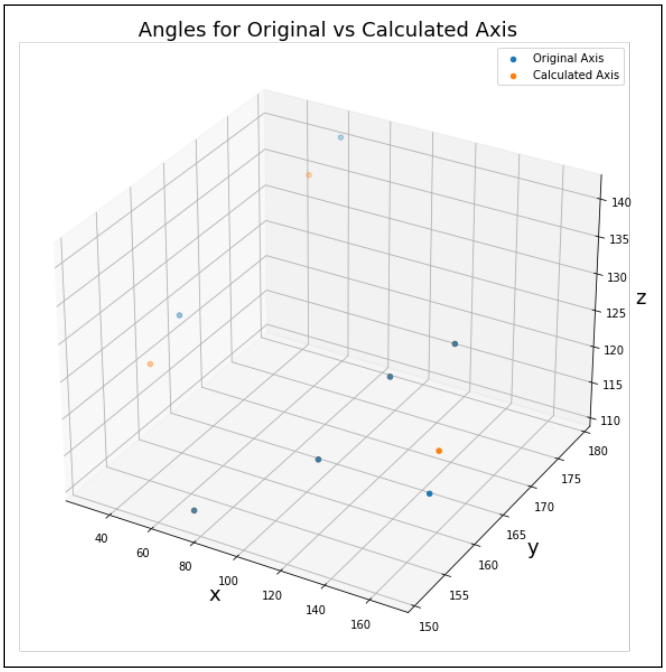

Figure 8: Angles calculated for real and calculated axis

Based on observed data from the figure 7, it was found that the percentage error on x axis was 0 and the percentage error on y axis was 1.913.

5. Conclusion

In conclusion, the objective of this research is to develop a method for robotic arms that employs inverse kinematics and image processing to recognize colored and shaped objects. The method is based on the Python programming language and OpenCV, a top image processing library. According to the study, color processing performs best in circumstances with consistent lighting, but it may not be accurate when there is a complicated background or dynamic lighting. In these situations, using CHT to follow the target object should be the main focus of the object detection strategy. The use of linear regression using data gathered from several trials has been shown to be an effective method for determining constant values with little error. The authors advise adding an extra degree of freedom for the end effector to reduce y-axis inaccuracy and changing all joints other than the end effector's servo motors to stepper motors in order to expand the workspace. The research highlights the potential for industrial applications, suggesting the increase in size and strength of the robotic arm, and the improvement of the camera used in the system for higher quality image/video. Furthermore, the use of multiple cameras can provide a 3D view of the system, which can improve the overall performance of the robotic arm. Overall, the proposed project has the potential to reduce human effort in tasks requiring high accuracy and precision, and can reduce the requirement of the human workforce to perform repetitive tasks.

Acknowledgments

The authors would like to express utmost appreciation to the Department of Electronics & Computer Engineering and Robotics Club at Purwanchal Campus for their invaluable input to this work. Their guidance, expertise, and support were crucial in the successful completion of this research. Additionally, the authors express their kind gratitude to colleagues and friends for their unwavering encouragement and support throughout the duration of the research.

References

[1] theGHIZmo. *EEZYbotARM Mk2 - 3D Printed Robot*, 2017.

[2] S. Bindu, Sankar Prudhvi, and N. Raja Sekhar G. Hemalatha. Object detection from complex background image using circular hough transform. *International Journal of Engineering Research and Applications*, 4(4):23–28, 2014.

[3] R. Hussin, M. Rizon Juhari, Ng Wei Kang, R.C. Ismail, and A. Kamarudin. Digital image processing techniques for object detection from complex background image. *Procedia Engineering*, 41:340–344, 2012. International Symposium on Robotics and Intelligent Sensors 2012 (IRIS 2012).

[4] Lennon Fernandes and B. R. Shivakumar. Identification and sorting of objects based on shape and colour using robotic arm. *2020 Fourth International Conference on Inventive Systems and Control (ICISC)*, pages 866–871, 2020.

[5] Muhatasim Intisar, Mohammad Monirujjaman Khan, Mohammad Rezaul Islam, and Mehedi Masud. Computer vision based robotic arm controlled using interactive gui. *Intelligent Automation & Soft Computing*, 27(2):533–550, 2021.

[6] Wong Guan Hao, Yap Yee Leck, and Lim Chot Hun. 6-dof pc-based robotic arm (pc-roboarm) with efficient trajectory planning and speed control. *2011 4th International Conference on Mechatronics (ICOM)*, pages 1–7, 2011.

[7] Van Rossum, Guido, Drake, and Fred L. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[8] Bradski and G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[9] Harris, Charles R., and Millman et.al. Array programming with NumPy. *Nature*, 585:357–362, 2020.

[10] Virtanen, Pauli, and Gommers et.al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[11] Raspberry pi Foundation. *Raspberry pi documentation*.

[12] Hunter and John D. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(3):90–95, 2007.