CS 3301 - REPORT

Hand Gesture Recognition System

By Raj Patel & Devang Donda

Introduction

The project focuses on developing a hand gesture recognition system with the help of OpenCV for hand detection and Teachable Machine (Web-based tool) for gesture recognition. The project focuses on two main stages: Processing the Image (Hand Detection) and Classifying the Image (Gesture Recognition). Initially, HSV was used for Image processing but was transitioned to Ostu's Binarization method for thresholding. For Gesture Recognition, two possible solutions were explored which are the convex hull method and machine learning; machine learning was chosen due to its numerous advantages.

Why Hand Gesture Recognition System?

1. Enhanced User Interface - Transforms the way users interact with technology. The idea of extending beyond the confinements of touchscreen offers a more natural and efficient way of controlling our devices. Imagine controlling TV without the need for a remote, scrolling through web pages without the need for touchpads and a mouse.
2. Inclusive Communication - One of the most known applications of the Hand Gesture recognition system is its ability to bridge the communication gaps for the impaired community which is done by translating gestures from sign language to text for those who do not understand the sign language.
3. Application Versatlity - The hand gesture recognition system is used in various sectors including the Apple's Vision Pro or the Meta Quest 3 for VR. In Virtual Reality, the method of interaction with the device without the need for controllers held in hand makes it an immersive experience. It can also be used in automobiles, security systems of smart homes and so on.

Hand Detection

Hand detection is a critical step in the process of developing a hand gesture recognition system. An effective hand detection algorithm is an algorithm that accurately segments the hand by the method of background subtraction regardless of the environment, which is later used for gesture classification.

Possible Solution: HSV Color Space Controlling Trackbars

Incorporating track bars in the hand gesture recognition for users to adjust HSV (Hue, Saturation, Value) values manually offers a tailored approach to hand gesture recognition. This customization is particularly beneficial in diverse lighting conditions and backgrounds, as it allows the user to fine-tune the system for optimal hand detection.

Advantages of HSV Trackbars

1. Enhanced Flexibility and Accuracy: Users can adjust the HSV values to suit specific lighting conditions and backgrounds, ensuring the system captures a clear image of the hand. This flexibility can significantly enhance the accuracy and precision of gesture recognition.
2. User Control and Customization: Providing users with the ability to manually adjust these settings empowers them to customize the system according to their unique environment. It is particularly useful in environments where lighting conditions are variable and unpredictable.
3. Improved Robustness in Diverse Conditions: By tweaking HSV values, the system can be made more robust and effective in various scenarios, such as at different times of the day or in an artificially lit environment.

Drawbacks of HSV Trackbars

1. Lack of Automation: The primary drawback is the absence of automation. Users need to manually adjust the settings each time they change their environment, which can be time-consuming and might require a learning curve to understand the optimal HSV settings.
2. User Dependency: The effectiveness of the system heavily relies on the user's ability to accurately adjust the HSV values. This can be challenging for users without prior experience or understanding of how HSV values influence image capture and processing.
3. Inconsistency in User Experience: Different users might set the track bars differently under similar conditions, leading to inconsistency in the user experience. This variability can affect the system's overall reliability and user satisfaction.
4. Potential for Error: Manual adjustments are prone to human error. Users might set the values in a way that degrades the system's ability to accurately recognize gestures, especially if they do not have a clear understanding of how these settings affect the image processing.

Conclusion:

While manual HSV adjustments via track bars offer greater control and customization, introducing automated, adaptive HSV value adjustments could significantly enhance the user experience by reducing manual effort and ensuring consistent performance across varied environments. This could involve using machine learning algorithms or adaptive image processing techniques that automatically adjust to changing lighting and background conditions, thereby maintaining the system's accuracy without requiring constant user intervention.

Otsu's Binarization Method

Otsu's Binarization method is an advanced technique of global thresholding that is used on grayscale images. It works by examining and calculating a threshold that minimizes the weighted within-class variance of black and white pixels respectively.

It has application in Bi-Modal Images, images whose histograms have two distinct peaks.

Original Image | Threshold Image | Histogram

Advantages of Otsu's BInarization Method

1. Automatic Threshold Calculation - Eliminated the need for manually calculating threshold values for a grayscale image which is very crucial for dynamic environments.
2. Foreground Separation - Effectively separates hand (foreground) from the rest of the image (background) under decent lighting conditions and simple background (background colors like white, blue, green, etc.),
3. Creation of a mask - The thresholding acts like a mask that isolates the hand which can be processed further.

Drawbacks of Otsu's BInarization Method

1. Dependence on Lighting Conditions - If an image has variable lighting, Otsu's method might not be able to find any single threshold for the image.
2. Foreground-Background Contrast - The algorithm can struggle to segment the foreground if the background has similar intensity values as the foreground.

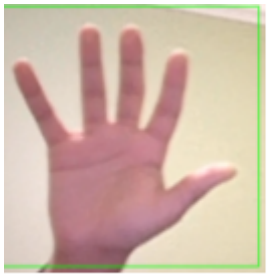
3. Non-Uniform Objects - If the object has varying intensities. Otsu’s binarization could incorrectly classify parts of objects as background or vice versa. This is because it tries to maximize variance between two classes
4. Bi-Modal Images: This method is highly suitable for images with two distinct peaks to calculate the threshold and if the image does not hold the requirements then this method fails to perform well.

Conclusion

While Otsu's Binarization Method is particularly useful due to its simplicity and effectiveness in ideal conditions, it struggles to perform well in uncontrolled environments. Its limitations must be considered carefully for practical applications like hand gesture recognition. Variations in light, noise and less contrast can lead to inaccurate thresholding affecting the ability of proper gesture recognition to take place.

Hand Detection: Processing the image

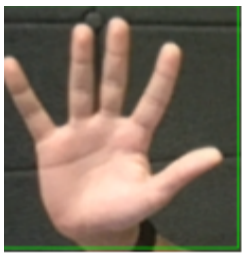
1. Image Acquisition - The process begins with capturing the video frame that is horizontally flipped to create a mirror image.
2. Image Cropping and Grayscaling- The video feed is cropped to focus on a specific region of interest achieved by drawing a rectangle where the hand is placed and the cropped image converted to Grayscale.
3. Gaussian blur - Applied to reduce the noise and the detail for smoother thresholding as the noise could particularly interfere with the detection of hand.
4. Thresholding - Otsu's Binarization method is used for thresholding the image to create a binary image in which the hand would be represented in white and the background in black.



Original Image

Threshold Image

5. Color Inversion Check
    - Bitwise Not Operation - To ensure that the hand is always white and the background is black, the mean color value is evaluated. If the hand appears light in front of a darker background (i.e., black or dark blue background), then bitwise not is applied to invert the pixel values.



Original Image

Threshold Image

Bitwise Not

- Morphological Operations - Erosion followed by dilation is applied (also called opening) which is particularly useful to separate two fingers that sometimes seem to be connected in a darker background.

Processed Images vs Raw Images

1. Consistency in Recognition - When a machine learning model trains on processed images, recognition is quite effective because processed images account for lighting conditions, dynamic backgrounds and skin tones. This consistency ensures that the model focuses on gestures required for recognition rather than irrelevant variations like lightning or background.
2. Emphasis on Features - Since unwanted data like backgrounds and lightning conditions are eliminated, the model focuses more on the shape and the features of the hand therefore increasing the chances of recognizing a gesture correctly.
3. Reduced Training Data - When a model is trained with raw sample images, the model must account for all possible gestures regardless of lighting or background conditions. Since processed images do not consider it, the size and complexity of training data are significantly reduced while keeping the accuracy high.

Gesture Recognition: Classifying the Processed Images

Machine Learning

Collecting sample images

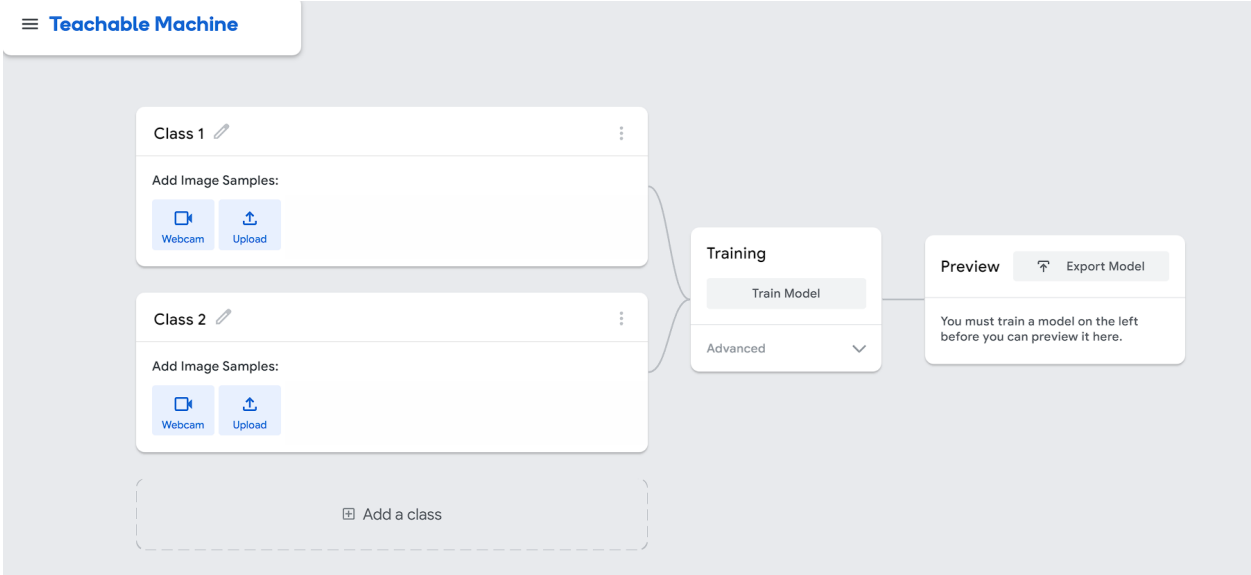
To recognize gestures, machine learning models have to be trained on processed sample images. Sample Images for training data should cover each gesture in different background settings, lighting conditions, and different shapes and sizes of hand.

For the project, sample images were collected in different background settings and lightning conditions but due to constraints only one set of hands was used. Approximately 80 - 200 images were used for each gesture.

Training the model with Teachable Machine

Teachable Machine is a web-based tool to create machine learning models quickly without any experience or know-how.

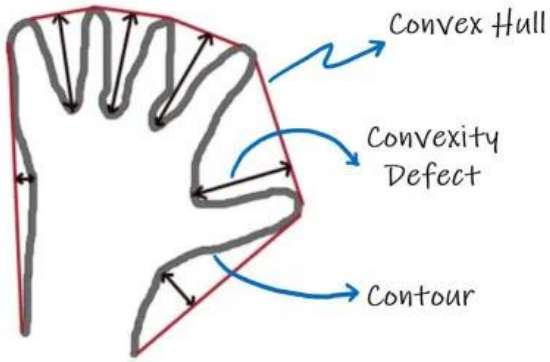
To get started, visit the website https://teachablemachine.withgoogle.com/, start a new project, and for each separate gesture, upload the sample processed images in each class (also label the gestures) and train the model with default settings which can be later exported and used with the help of tensorflow library in python.



Possible Solution: Convex Hull and Convexity Defects

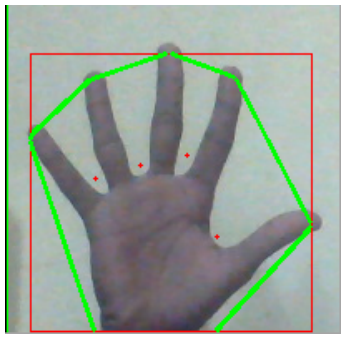
Theory

Convex Hull in the context of image processing is a concept borrowed from mathematics and geometry. It is the smallest convex set that encloses a given shape or set of points. The internal angles of a convex polygon are less than 180 degrees. So, if there exists any internal angle greater than 180, that is a convexity defect, which is used to count gaps between 2 fingers.



- Imagine stretching a Rubber Band: Picture the points (or the shape) you are considering, like the contour of a hand in an image. Now visualize stretching a rubber band so that it encompasses all these points. When you release the rubber band, it snaps around the points, forming a tight boundary. The boundary is the convex hull.
- Characteristics of Convexity: A shape is convex if, for every pair of points within the shape, the line segment connecting them lies entirely within the shape. The convex hull does not have any indentations or 'inward' curves relative to the points it encloses.
- Relevance in Image Processing: In hand gesture recognition, the convex hull is used to simplify the representation of the hand. By creating a convex boundary around the hand's contour, one can more easily identify and analyze features like the gaps between fingers (convexity defects), which are crucial for interpreting gestures.

In summary, a convex hull is a tool in image processing, providing a simplified and convex representation of shapes, which is particularly useful in applications like gesture recognition.

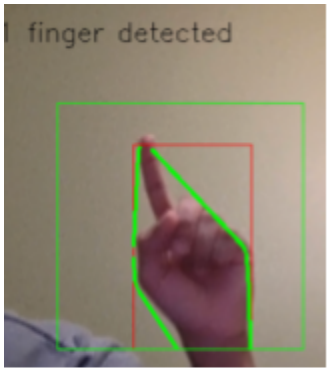
- 

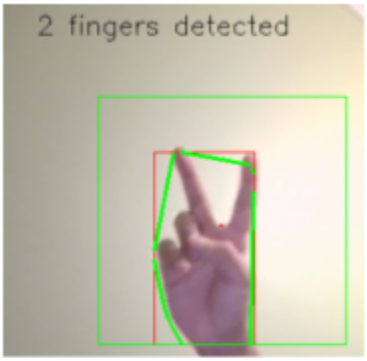
  Red dots between fingers represent convexity defects and the green boundary is the convex hull.

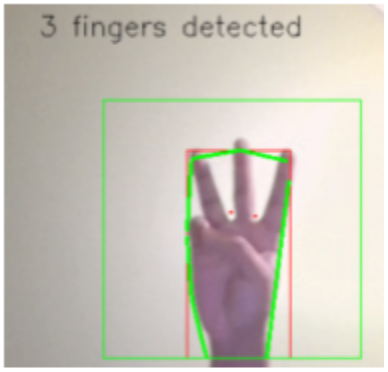
Working: Usage of Convex Hull in OpenCV

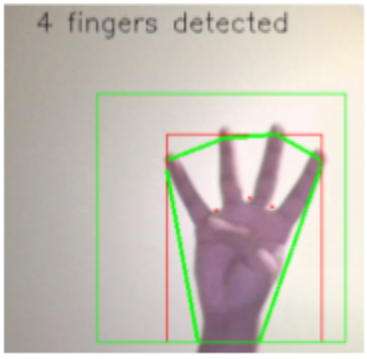
1. Image Acquisition - The process begins with capturing the video frame that is horizontally flipped to create a mirror image.
2. Hand Segmentation: This is done by detecting skin color detection or background subtraction methods.
3. Thresholding and Contour Detection: Thresholding is applied to create a binary image where the hand is separated from the background. The contours of the hand are then detected in this binary image. OpenCV functions like findContours are used for this purpose.
4. Applying Convex Hull: The Convex Hull algorithm is applied in OpenCV which is done using the convexHull function. The convex hull of a shape is the tightest convex shape that completely encloses the shape. For a hand, this outlines the hand while ignoring indentations between the fingers.
5. Gesture Recognition: The resulting convex hull can then be analyzed to recognize gestures which is done by looking at the characteristics of the hull such as the number of convexity defects (spaces between fingers) and the orientation of the hull. Simple gestures can be recognized based on these features, such as counting the number of fingers on the screen = number of defects + 1 (Not always).

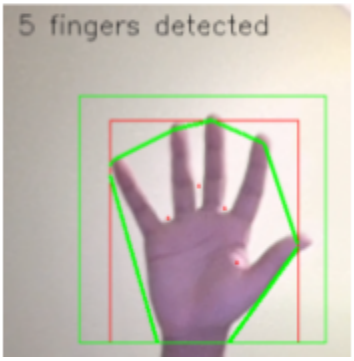
Finally, the recognized gesture is outputted for further action or response in the form of a visual display.









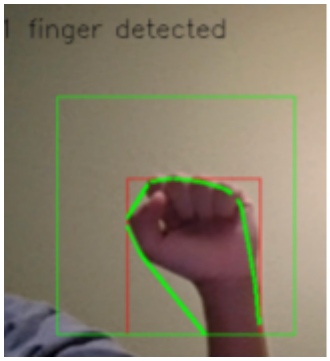


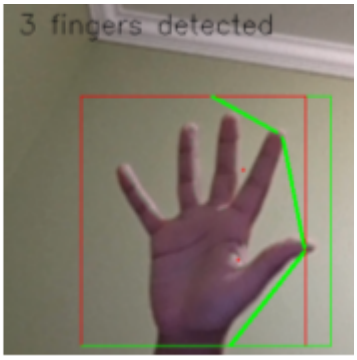
Advantages of the Convex Hull Method

1. Easy to implement: Implementing this method using libraries like OpenCV in Python is straightforward, making it accessible for those new to computer vision.
2. Efficiency in Processing: The Convex Hull algorithm is computationally efficient. It can process images quickly, making it suitable for applications where real-time gesture recognition is needed.
3. Effectiveness with Basic Gestures: For basic hand gestures, especially those with clear and distinct shapes, Convex Hull can be quite effective. It works well when the gestures form easily discernible convexity defects.
4. Foundation for More Complex Analysis: Convex Hull could potentially serve as a starting point for more complex gesture recognition systems. It could be used in combination with machine learning algorithms to improve accuracy and handle more complex gestures.

Drawbacks of the Convex Hull Method

1. Inability to Detect Non-Convex Gestures: Convex Hull relies on finding convexity defects (gaps between fingers). If a gesture forms a non-convex shape, like a hand making a fist, or fingers clasping the thumb, the method may fail to recognize it due to the lack of detectable convexity.
2. Dependence on Angle and Orientation of the Hand: The recognition accuracy heavily depends on how the hand is positioned relative to the camera. If the hand is tilted or turned in a way that reduces visible convexity, the method might incorrectly interpret the gesture.
3. Difficulty in Differentiating Similar Gestures: Gestures with similar external contours but different internal configurations might be indistinguishable using Convex Hull. This is because it primarily focuses on the outer boundary and may miss nuances in finger positioning.
4. Limited to Static Gesture Recognition: The method is more suitable for recognizing static gestures. Dynamic gestures involving movement or changes over time are challenging to capture and interpret using only Convex Hull.





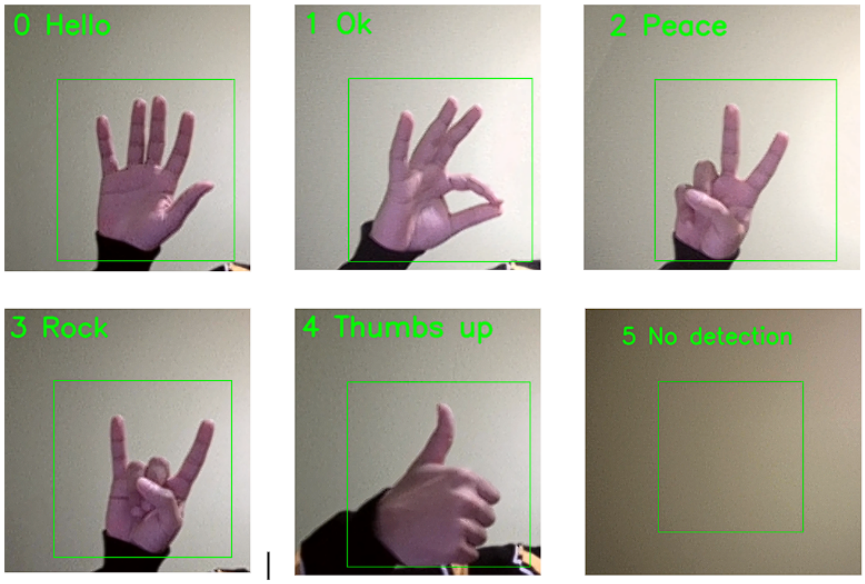
Empirical Analysis: Testing & Validating the Solution

Experimental Setup

Experiments were conducted using a 5 mp front camera of an HP Envy x360 laptop. Training and testing were done in two different environments: Indoors and outdoors which are the Bedroom and Patio respectively. The environments were chosen to evaluate the solution's performance under distinct lighting conditions. Indoors offered dim and controlled lighting whereas outdoors offered natural and dynamic lighting conditions.

Experiments were conducted within the region of interest (rectangle frame), approximately 30-45 centimeters from the camera. This distance was considered optimal because of the constraints of image processing techniques applied and limitations of the front camera's resolution and focus for capturing the necessary detail required for processing the image.

The machine learning was trained on default settings on teachable machine for 5 gestures including hello, ok, peace, rock, thumbs up and no detection.



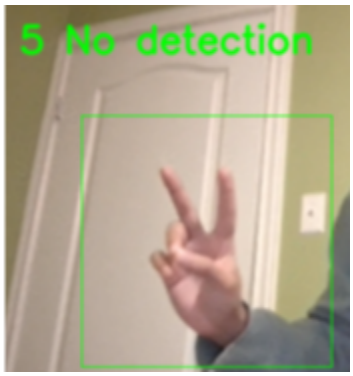
Under controlled lighting conditions and a perfect background, the results are highly accurate but the inconsistency arises when either the lighting condition or background is not ideal, which affects the model's ability to recognize gestures.

Challenges in training the Machine Learning Model

Initially, the model was not able to differentiate between ok and hello when the angles or the position of the hand was slightly changed. To address this, the more sampled processed images of both gestures were collected for the model to train on thereby, increasing its ability to recognize gestures correctly.

Assessing the performance and efficiency of Image Processing

Image 1

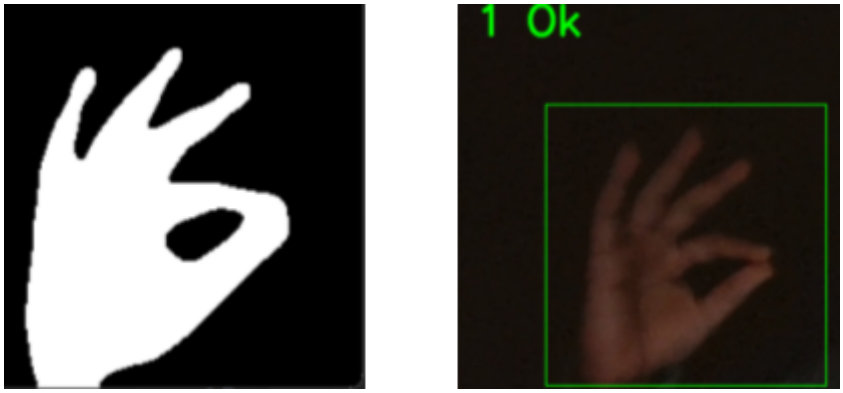


No detection occurs because Otsu's Binarization relies on a global threshold to separate the hand from the background. If there exist shadows and highlights, the threshold is not calculated properly and parts of the hand are classified as background.

Image 2

The gesture is supposed to be a “peace” symbol however the potential issue why it could not be detected correctly is the background does not contrast well enough with the hand and is also of similar color to the hand, the hand is overexposed and the background is complex.

Image 3

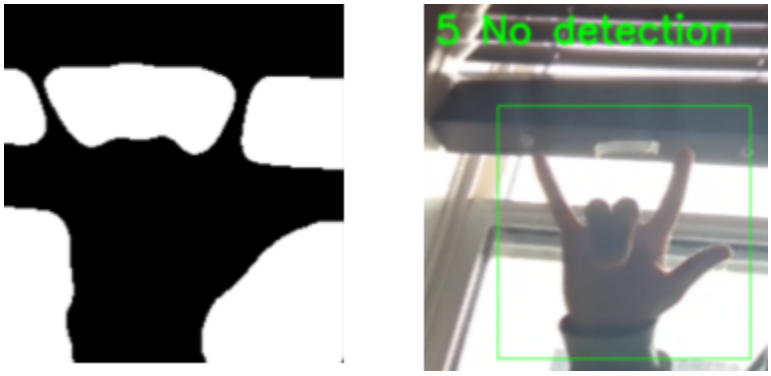


The gesture “okay” is recognized by the model because the image is preprocessed accurately due to the foreground being distinct from the background, the background has a distinct color, and the hand is well-lit by the screen of the laptop.

Image 4

The gesture is recognized correctly even though the background is a bit complex because the foreground has a very strong light source falling on it which cause the histogram of the grayscale image to have two distinct peaks (bi-modal image)

Image 5



The gesture “rock’” is not recognized because there is a strong light coming from the background (window) complicating the process of thresholding and the parts of the window are considered as foreground.

Image 6

This is another example of the gesture being recognized perfectly under perfect natural light conditions. (The background is the sky)

Conclusions

The gesture recognition is handled flawlessly by the machine learning model. However, the the model's restriction is only the amount of sample processed images the model is trained on which decreases its ability to recognize gestures with high accuracy.

It can be concluded that even though image processing works well in a decent environment under controlled lighting conditions, it has to be improved to accurately process the images in rough environments where the lighting is not regulated.

The image processing is highly sensitive to lighting conditions, shadows, highlights, under-exposure, and over-exposure which could lead to inaccurate thresholding. One possible scenario to handle this is by automatic dynamic range adjustment to control the sensitivity of the light.

The image processing also does not work well if there is no clear contrast between the foreground and the background. Histogram equalization techniques could help enhance the contrast between foreground and background.

Images that have non-uniform illumination could use adaptive thresholds that could potentially calculate threshold accurately.

References

Sana'a Khudayer Jadwa. “Otsu Segmentation Method for American Sign Language Recognition.” *International Journal of Engineering Research and General Science*, vol. 3, no. 5, 2015, pnrsolution.org/Datacenter/Vol3/Issue5/117.pdf. Accessed 5 Dec. 2023.

Xu, Yanan, et al. “Hand Gesture Recognition Based on Convex Defect Detection.” *International Journal of Applied Engineering Research*, vol. 12, 2017, pp. 7075–7079, www.ripublication.com/ijaer17/ijaerv12n18_04.pdf. Accessed 6 Dec. 2023.

Renotte, Nicholas. “Real Time Sign Language Detection with Tensorflow Object Detection and Python | Deep Learning SSD.” *Www.youtube.com*, 5 Nov. 2020, www.youtube.com/watch?v=pDXdlXlaCco&t=390s. Accessed 5 Dec. 2023.