AN ENHANCEMENT OF HAAR CASCADE ALGORITHM APPLIED TO FACE RECOGNITION FOR GATEPASS SECURITY

A Thesis Presented to the
Faculty of Computer Science Department
College of Information Systems and Technology Management
Pamantasan ng Lungsod ng Maynila

In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Science

By
Clarence A. Antipona
Romeo R. Magsino III

Thesis Adviser

Prof. Raymund M. Dioses

October 2024

APPROVAL SHEET

The thesis hereto titled.

AN ENHANCEMENT OF HAAR CASCADE ALGORITHM APPLIED TO FACE RECOGNITION FOR GATEPASS SECURITY

Prepared and submitted by Clarence A. Antipona and Romeo R. Magsino III in partial fulfilment of the requirements for the degree of Bachelor of Science in Computer Science has been examined and is recommended for acceptance and approval for **ORAL EXAMINATION**.

Prof. Raymund M. Dioses
Adviser

PANEL OF EXAMINERS

Approved by the Committee on Oral Examination
with a grade of __________ on ________.

____
Panel Chair
Chairman

_________________________	_________________________
Panel Member	Panel Member
Member	Member

Accepted and approved in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science

_________________________	_________________________
Name	Name
Chairperson, CS Department	Dean, College of Information Systems, and Technology Management

ABSTRACT

This study is focused on enhancing the Haar Cascade algorithm to decrease the false positive and false negative rate in face matching and improve face detection accuracy even under challenging conditions. The face recognition library from OpenCV was implemented with Haar Cascade where 128-dimensional vectors representing the unique features of a face were encoded. A subprocess was applied where the grayscale image from Haar Cascade was converted to RGB to improve the face encoding. Logical process and filtering were used to decrease non-face detection.

The Enhanced Haar Cascade Algorithm produced a 98.39% accuracy rate, 63.59% precision rate, 98.30% recall rate, and 72.23% in F1 Score. In comparison, the Haar Cascade Algorithm achieved a 46.70% - 77.00% accuracy rate, 44.15% precision rate, 98.61% recall rate, and 47.01% in F1 Score. Both algorithms used the Confusion Matrix Test with 301,950 comparisons using the same dataset of 550 images. The 98.39% accuracy rate shows a significant decrease in false positive and false negative rates in facial recognition. Face matching and face detection are more accurate in images with complex backgrounds, lighting variations, and occlusions, or even those with similar attributes.

Thus, the Enhanced Haar Cascade Algorithm is more reliable in facial recognition.

ABSTRACT

This study is focused on enhancing the Haar Cascade algorithm to decrease the false positive and false negative rate of face recognition in images with variations in lighting, facial expressions, and occlusions to increase accuracy. The face recognition library was applied with Haar Cascade where 128-dimensional vectors representing the unique features of a face were encoded. The Enhanced Haar Cascade Algorithm produced a 98.39% accuracy rate, in comparison, the Haar Cascade Algorithm achieved a 46.70% - 77.00% accuracy rate. Both algorithms used the Confusion Matrix Test with 301,950 comparisons using the same dataset of 550 images. The 98.39% accuracy rate shows a significant decrease in false positive and false negative rates in facial recognition in images with complex conditions.

ABSTRACT

This study is focused on enhancing the Haar Cascade algorithm to decrease the false positive and false negative rate of face matching in images that share similar attributes to increase accuracy. A subprocess was applied where the grayscale image from Haar Cascade was converted to RGB to improve the face encoding extraction. The Enhanced Haar Cascade Algorithm achieved an accuracy rate of 99.58%, a 22.58% increase compared to the 46.70% - 77.00% baseline accuracy rate. Both algorithms used the Confusion Matrix Test with 301,950 comparisons using the same dataset of 550 images. The 99.58% accuracy rate shows a significant decrease in false positives and false negatives in face matching with images that share similar attributes.

ABSTRACT

This study is focused on enhancing the Haar Cascade algorithm to decrease the false positive and false negative rate of face detection to detect real human faces accurately. A logical process and filtering were applied to decrease non-face detection, iterating the parameters and selecting the face that meets the criteria. The Enhanced Haar Cascade Algorithm produced a 97.19% accuracy rate, an 8.46% increase, in comparison, the Haar Cascade Algorithm achieved an 88.73% accuracy rate. Both algorithms used the Confusion Matrix Test with 301,950 comparisons using the same dataset of 550 images. The 97.19% accuracy rate shows a significant decrease in false positive and false negative rates in face detection and can detect real human faces.

ACKNOWLEDGEMENTS

I, **Clarence A. Antipona**, and **Romeo R. Magsino III**, would like to express our heartfelt gratitude:

The **Almighty God**, who strengthens and blesses us in our daily lives, providing guidance and resilience even in the midst of challenges.

Our **Parents** and **Guardians**, who always guide us with unwavering love and support, serving as our pillars of strength and inspiration. Their constant encouragement has been the bedrock of our perseverance and success.

Dr. Khatalyn E. Mata, our Thesis 1 professor, who enriched us with profound knowledge and expertise in research, and consistently encouraged us to strive for excellence.

Dr. Dan Michael A. Cortez, our Research Writing professor, who guided us meticulously to ensure our work meets the highest academic standards and fostered our growth as competent researchers.

Prof. Raymund M. Dioses, our Thesis Adviser, who tirelessly assessed our progress, ensuring that quality and standards are upheld in our research endeavors, and whose dedication and patience have been invaluable.

And lastly, to our **Friends** and **Special Someone** who give us boundless inspiration and unwavering support. Their encouragement and understanding have been crucial, providing the emotional and moral support needed to complete this journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter One

INTRODUCTION

1.1 Background of the Study

Facial recognition is increasingly common in various applications ranging from security systems to social media platforms. The effectiveness of such systems hinges upon the robustness of the underlying algorithms employed for facial recognition. Haar Cascade algorithms have been widely utilized in this domain due to their computational efficiency and satisfactory accuracy. Haar Cascade Algorithm is a type of machine learning wherein a classifier is used from a great deal of positive and negative photos. Haar feature-based cascade classifiers are utilized for object detection purposes. This classifier employs a machine learning technique where a cascade operation is applied to images to detect objects in subsequent images. (Shetty et. al., 2021).

However, challenges persist in achieving reliable facial recognition under various conditions such as changes in lighting, facial expressions, and occlusions. A typical facial recognition doesn't go into the depth of the image, it just checks for the relevant similarities. This can sometimes become a vulnerability to bypass this kind of system (Minu, et. al, 2020). Before the introduction of the Haar-cascade algorithm in 2001, several object recognition applications were developed. Additional principal component analysis (PCA) was utilized to reduce the complexity of face images, decrease data size, and eliminate noise. PCA was also integrated with neural networks for face recognition and sex determination. However, these earlier algorithms exhibited certain drawbacks, such as a low classification percentage (ranging from 31.48% to 94.5%) and high mean square error (ranging from 0.02 to 0.12).

Meanwhile, the Haar cascade algorithm offered advantages such as quick detection and high efficiency, leading to its widespread application in numerous studies. It was employed for face detection in registered individuals from input video streams, demonstrating high accuracy and speed. Nevertheless, the effectiveness of this implementation relied on factors like video quality (illumination, angle, obstacles)

A refinement of the Haar cascade algorithm was proposed by Cuimei et al. (n.d.), which combined three different classifiers: color HSV, histogram matching, and eyes/mouth detection. This enhanced algorithm was subsequently utilized by Arreola et al. in 2018, who applied it to a quad-rotor Unmanned Aerial Vehicle (UAV) for face recognition purposes. In addition to the Haar cascade algorithm, other approaches can be applied to real-time tracking tasks, including local binary patterns (LBPs) or histogram of oriented gradients (HOG). A comparative study involving three algorithms—Haar cascade, LBP, and HOG—was conducted for object detection using UAVs. The results indicated that the Haar-like cascade outperformed LBP in accuracy rate and was faster than HOG (Phuc et. al., 2019)

In this study, we will implement a combination of a Haar cascade algorithm and the face recognition model from OpenCV. The outcome of this implementation will be an enhanced facial recognition system, titled " An Enhancement of Haar Cascade Algorithm Applied to Face Recognition for Gate pass Security" which will significantly improve security by providing accurate and efficient identification of individuals, thereby preventing unauthorized access and ensuring the safety of individuals.

1.2 Haar Cascade Algorithm Pseudocode

```
STEP 1:
    1.1 Load images using OpenCV's imread function.
STEP 2:
                                                          SOP 2
    2.1 Convert image to grayscale.
    2.2 Detect faces using Haar cascade classifier in grayscale images. (scaleFactor,
    minNeighbors, and minSize are used for face detection.)
    2.3 Put boxes in the detected faces                   SOP 3
STEP 3:
    3.1 Convert faces to LBPH.
    3.2 Extract the face region from the grayscale image using the detected face
    coordinates.
                                                          SOP 1
    3.3 Apply the Local Binary Pattern (LBP) transformation to the face region.
    3.4 Compute a normalized histogram of the LBP values.
STEP 4:
    4.1 Convert each faces region to an LBP histogram.
    4.2 Compute the Euclidean distance between the two LBP histograms.
    4.3 Convert the distance to a similar percentage
STEP 5:
    5.1 Show images and their similarity percentage
```

1.3 Haar Cascade Algorithm

Haar features utilize a cascading function along with cascading windows to detect features present within each window. This process involves classifying positive data points (regions containing the desired object, like faces) and negative data points (regions without the object) as the window moves across the input image. It aims to efficiently reject false or negative data points by adjusting hyperparameters, ensuring accurate detection. Additionally, feature computation and matching are performed independent of image scale or location during the classification process.

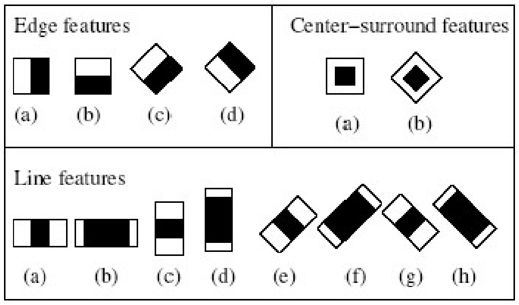


Figure 1 Haar Features

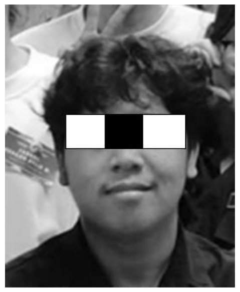


Figure 2 Haar Features Traversing an Image

Figure 1 shows the examples of Haar features for detecting faces in the image while Figure 2 visualizes the Haar features as it traverse to the face of the given image.

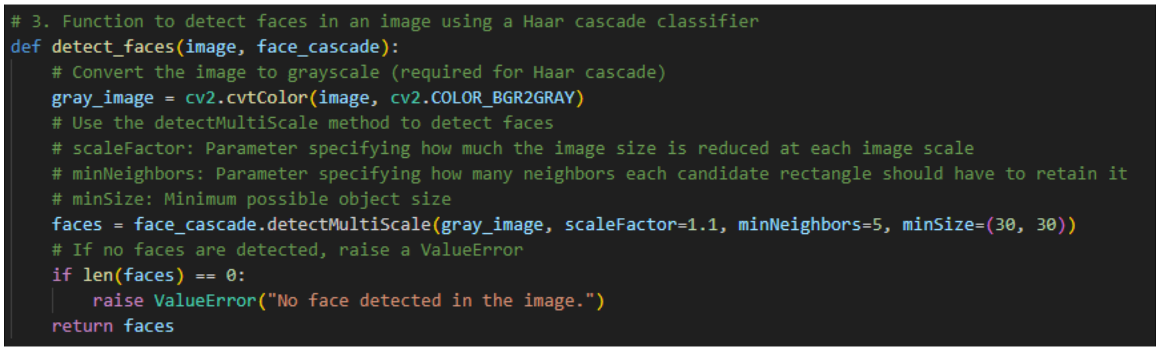
To employ the Haar Cascade Algorithm for face recognition, a XML file containing the pre-trained Haar Cascade is necessary (haarcascade_frontalface_default.xml). Once obtained, integrating it into the code enables the detection of faces in images. The crucial function for this task is detectMultiScale(), which scans an input image for objects of various sizes and returns a list of rectangles enclosing the detected objects, in this case, faces.

To fine-tune the face detection and recognition process, several parameters come into play:

Scale Factor: This parameter, usually set to 1.1, decides how much smaller each version of the image should be as we go through different sizes in the pyramid. (a series of images at various sizes. Think of it as creating multiple versions of the same picture, each smaller than the last.) This is important for finding faces of different sizes in pictures.

Min Neighbors: This parameter specifies the minimum number of neighboring rectangles required for each detected face. Its value profoundly impacts the quality of results. Higher values can lead to lower detection quality, whereas values typically ranging from 3 to 6 often yield optimal results.

Min Size: Used to define the minimum allowable size of a detected face.



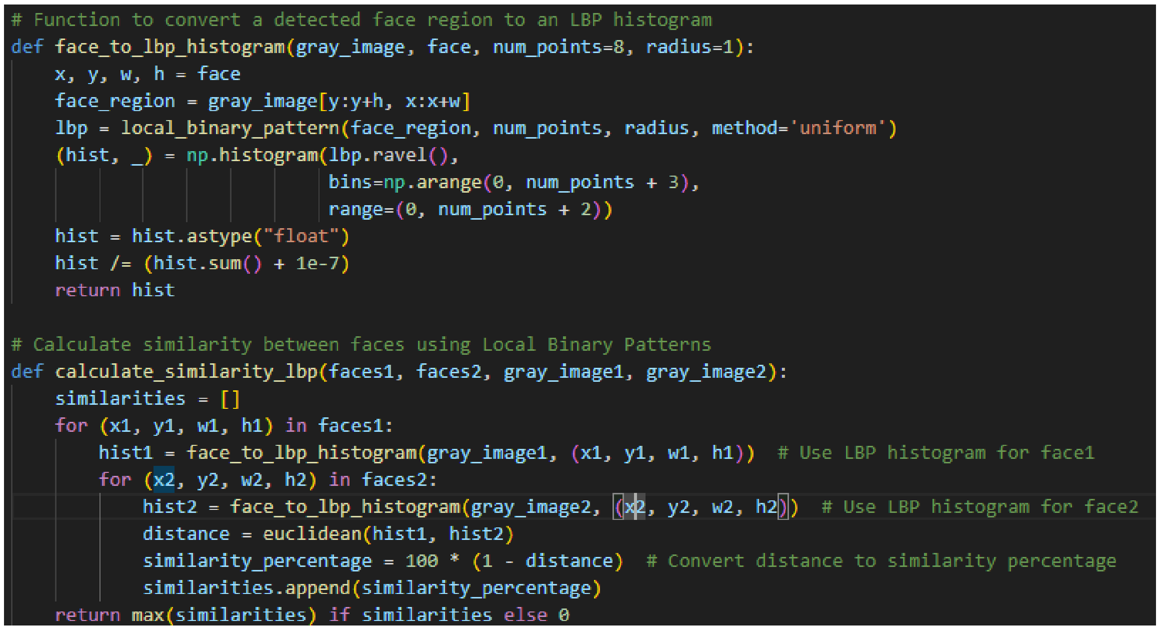
```
# 3. Function to detect faces in an image using a Haar cascade classifier
def detect_faces(image, face_cascade):
    # Convert the image to grayscale (required for Haar cascade)
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # Use the detectMultiScale method to detect faces
    # scaleFactor: Parameter specifying how much the image size is reduced at each image scale
    # minNeighbors: Parameter specifying how many neighbors each candidate rectangle should have to retain it
    # minSize: Minimum possible object size
    faces = face_cascade.detectMultiScale(gray_image, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))
    # If no faces are detected, raise a ValueError
    if len(faces) == 0:
        raise ValueError("No face detected in the image.")
    return faces
```

Figure 3 Haar Cascade Parameters (scaleFactor, minNeighbors, minSize)

Figure 3 shows the parameters of the Haar Cascade Algorithm. These parameters are utilized within the detect_faces() function, where the detectMultiScale() method is invoked with specific values. For instance, a scale factor of 1.1 and a minimum neighbor count of 5 are applied to efficiently detect faces in the given images.

After detecting faces in two images using the Haar Cascade classifier, the code proceeds to compare the detected faces for similarity. The detect_faces function first converts the input image to grayscale and then uses the Haar Cascade classifier to detect faces, returning the coordinates of the detected face regions. If no faces are detected, it raises a ValueError.

The detected faces are then cropped and converted to Local Binary Pattern (LBP) histograms. This conversion is handled by the face_to_lbp_histogram function, which takes a grayscale image and the coordinates of a face. It extracts the face region, applies the LBP transformation, and computes a normalized histogram of the LBP values.



```python
# Function to convert a detected face region to an LBP histogram
def face_to_lbp_histogram(gray_image, face, num_points=8, radius=1):
    x, y, w, h = face
    face_region = gray_image[y:y+h, x:x+w]
    lbp = local_binary_pattern(face_region, num_points, radius, method='uniform')
    (hist, _) = np.histogram(lbp.ravel(),
                             bins=np.arange(0, num_points + 3),
                             range=(0, num_points + 2))
    hist = hist.astype("float")
    hist /= (hist.sum() + 1e-7)
    return hist

# Calculate similarity between faces using Local Binary Patterns
def calculate_similarity_lbp(faces1, faces2, gray_image1, gray_image2):
    similarities = []
    for (x1, y1, w1, h1) in faces1:
        hist1 = face_to_lbp_histogram(gray_image1, (x1, y1, w1, h1))  # Use LBP histogram for face1
        for (x2, y2, w2, h2) in faces2:
            hist2 = face_to_lbp_histogram(gray_image2, (x2, y2, w2, h2))  # Use LBP histogram for face2
            distance = euclidean(hist1, hist2)
            similarity_percentage = 100 * (1 - distance)  # Convert distance to similarity percentage
            similarities.append(similarity_percentage)
    return max(similarities) if similarities else 0
```

Figure 4 Haar Cascade Code Snippet

Figure 4 shows the Haar Cascade Algorithm code snippet for both histogram and similarity calculations. The calculate_similarity_lbp function calculates the similarity between the detected faces in both images. For each face detected in the first image, it computes the LBP histogram and compares it to the LBP histograms of the faces detected in the second image. The similarity between two faces is determined by the Euclidean distance between their LBP histograms, which is then converted to a similarity percentage. The function returns the maximum similarity percentage found among all face pairs.

1.4 Statement of the Problem

The study seeks to address the problems of the Haar cascade algorithm in facial recognition.

1. **Haar cascade algorithm lacks accuracy in facial recognition, particularly in scenarios with variations in lighting, facial expressions, and occlusions.**

   (Singh, Herunde, Furtado, 2020) Stated that the major problem of face detection while using a Haar cascade is that the image contains both simple and complex background. (Ahmad et. al., 2021). In their study ***"Real time face recognition of video surveillance system using haar cascade classifier",*** they discovered that there are some problems due to the low lighting condition, degree of facial angle, and accessory covering the face which can result to a False Positive or False Negative.

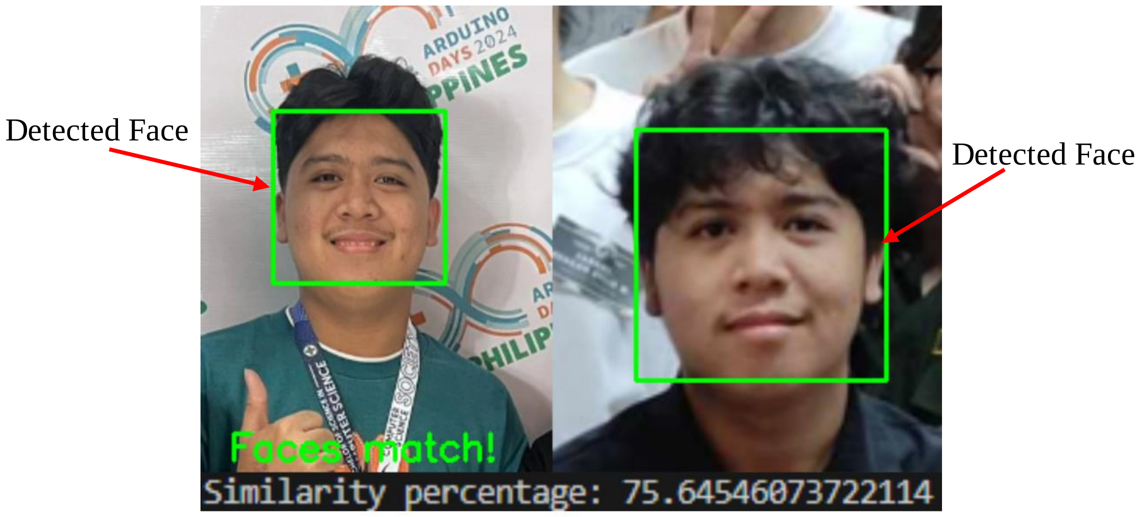


Figure 5 Similarity percentage of the same person

Figure 5 Shows a similarity percentage of 75.64% which turns out to be a good matching and a true positive.

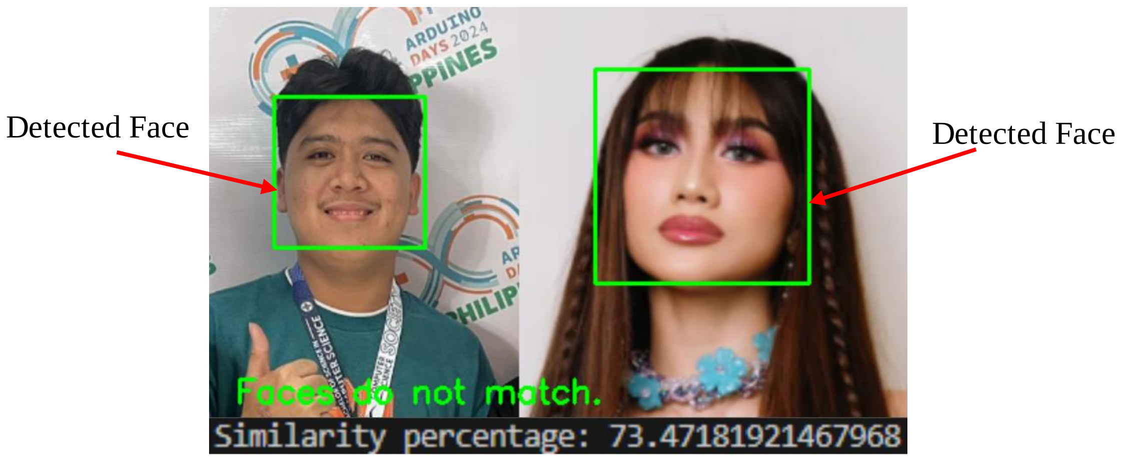


Figure 6 Similarity percentage (different person no.1)

Figure 6 Shows a similarity percentage of 73.47% which is a true negative in comparing two different facial images.

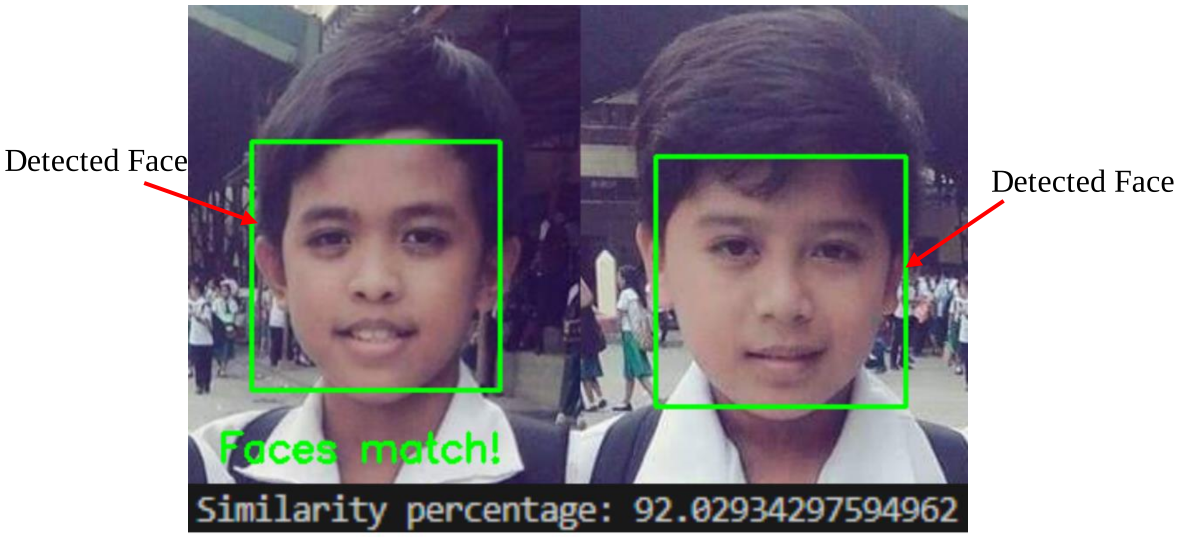


Figure 7 Similarity percentage (different person no.2)

Figure 7 Shows a similarity percentage of 92.02% in comparing two different people with almost the same facial feature, this poses a high percentage for false positives.

2. **Haar cascade algorithm struggle to differentiate between individuals sharing similar attributes (e.g., wearing glasses), leading to potential false positives.**

   According to (Adekola, 2023) Haar Cascade algorithm doesn't just work automatically, it might sometimes work with automatic settings for simple and flexible images but for complex images, detection or recognition requires a lot of manual tuning of hyperparameters, false-positive detection is higher, and it is also less accurate compared to deep-learning face detectors. In another study by (Chinimilli, 2020) They created a modified Haar Cascade algorithm with LBPH where they achieve a face recognition rate of 77% with 28% false positive rate. They also claim that their system can recognize students with glasses or beard, though is it's still posing a problem considering the 28% false positive rate.

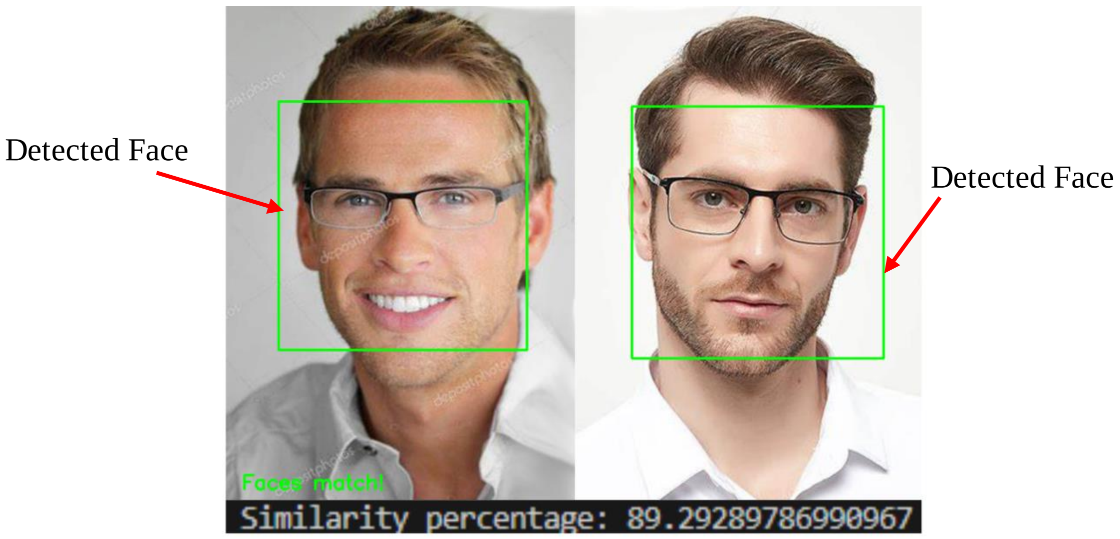
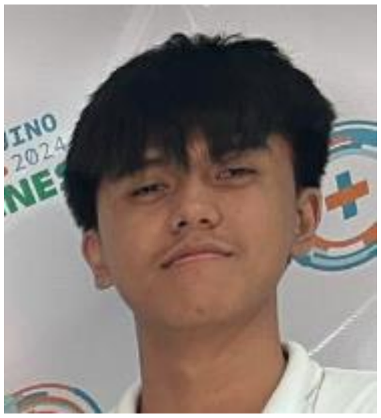


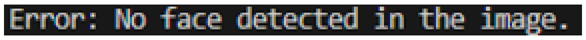
Figure 8 Similarity percentage of a different person (with glasses)

Figure 8 Shows that the algorithm matches the facial image with a similarity percentage of 89.29% (considering that that acceptance threshold is 75%) even though the two images are from different persons, thus the eyeglasses pose a conflict in the algorithm when the matching process occurs, this leads to a false positive result.

3. **Haar cascade algorithm faces some challenges in accurately detecting real faces (non-face), resulting in false negatives.**

In the study ***"Facial recognition using Haar Cascade and LBP classifiers"*** (Shetty et. al., 2021). They concluded that Haar Cascade classifier is much accurate than LBP classifier, but the disadvantage is sometimes it doesn't detect faces and most likely the faces of the children. False negatives occur when the algorithm fails to detect a face that is present in an image or video frame.





```
Error: No face detected in the image.
```

Figure 9 Error in facial detection

Figure 9 Shows an error message when trying to compare two images, one image is detected but the image of the person above cannot be detected with facial recognition.

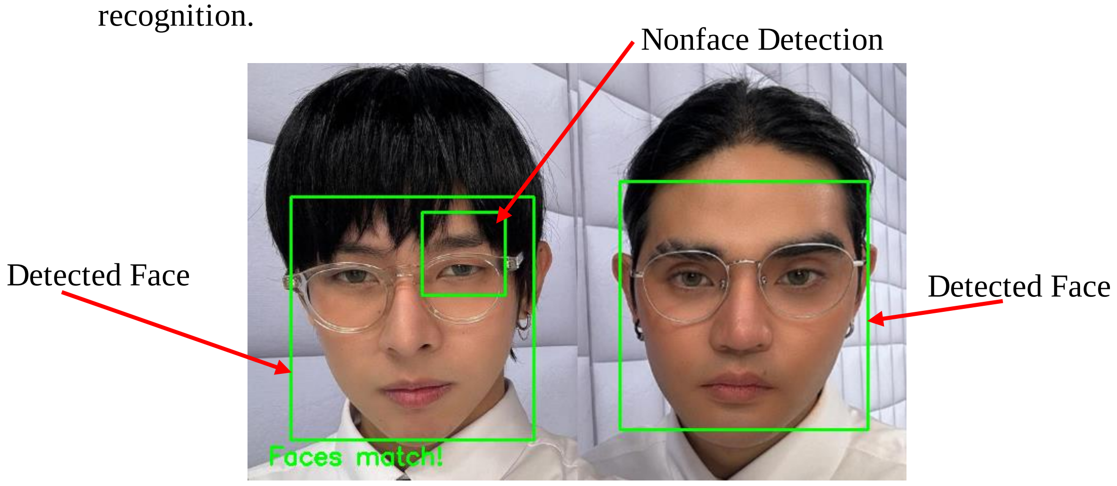


Figure 10 non-face detection

Figure 10 Shows a false positive face detection, the green boxes represent the area where a face is detected, hence it detected 3 faces base on the number of the green boxes, but we can see that there’s only 2 faces and the detected one is an eye.

1.5 Objective of the Study

The general objective of the study is to improve facial recognition accuracy and reliability of the Haar cascade algorithm to increase the University's security. The researchers aim to achieve these objectives:

1. To enhance the Haar Cascade Algorithm that accurately recognizes facial features under challenging conditions like variations in lighting, facial expressions, and occlusions.

2. To enhance the algorithm that is more reliable in facial matching in faces that share similar attributes.

3. To enhance the algorithm that detects real human faces

1.6 Significance of the Study

This study aims to improve/modify the Haar Cascade algorithm, specifically targeting improvements in facial recognition accuracy. By enhancing the existing algorithm through the integration of classifiers or the exploration of alternative algorithms, the research seeks to elevate the performance, reliability, and efficiency of Haar Cascade algorithm for facial recognition systems, particularly for the following stakeholders:

Developers/Programmers: For developers and programmers working on facial recognition applications, the enhanced Haar Cascade algorithm presents a valuable innovation. By incorporating additional classifiers or alternative algorithms, they can access improved tools for building more reliable and effective facial recognition systems.

Future Researchers: The findings of this study serve as a foundation for future research in the field of facial recognition. By documenting the modifications made to the Haar Cascade algorithm and evaluating their performance, this research provides insights and directions for further exploration. Future researchers that want to further improve this study can use this as their references.

Students: This study provides an educational opportunity to explore image processing techniques using low-end processors, in contrast to the high-resource image processing models.

1.7 Scope and Limitations

This study primarily focuses on enhancing Haar Cascade Algorithm to improve reliability in facial recognition, particularly in decreasing false positive rate and false negative rate. The development of a Enhanced Haar Cascade algorithm is tailored to effectively detect and recognize facial features under diverse environmental conditions by leveraging the capabilities of the face_recognition library and optimizing algorithmic parameters. It does not explore alternative facial recognition algorithms as some of it use large scale pre-trained data and high-end computation requirements. The effectiveness of the enhanced algorithm may be contingent upon the quality and diversity of the data used, potentially limiting its applicability across all scenarios. The study does not address the computational overhead associated with the integration of additional libraries or the potential increase in processing time required for facial recognition tasks. Additionally, the haar feature is different from haar cascade as haar feature is used in Viola and Jones Algorithm to create robust cascade classifier. The researchers won't create trained data, instead we will use existing pre-trained data, namely the haarcascade_frontalface_default.xml by Rainer Lienhart, 2000. The application of this study will be a Facial Recognition Attendance System for the University

1.8 Definition of Terms

Classifier: A classifier in machine learning is an algorithm that automatically orders or categorizes data into one or more of a set of "classes."

detectMultiScale: A function or method used in object detection, particularly in the context of the Haar Cascade Algorithm, which analyzes images or video frames at multiple scales to detect objects like faces.

Euclidean distance: A method that measures the similarity or dissimilarity between facial feature vectors extracted from images. Each face is represented by a unique feature vector that encodes key facial attributes, such as distances between landmarks or texture patterns.

face_recognition: A Python library for facial recognition that provides tools and algorithms to detect and recognize faces in images or videos.

face_encoding: A process used in facial recognition systems to convert the unique features of a face into a numerical representation, typically a vector of floating-point numbers.

Facial recognition: A technology that identifies or verifies individuals by analyzing and comparing patterns based on their facial features.

False negative: A result indicating that a particular condition or characteristic is absent when it is present, typically in the context of facial recognition, referring to instances where a person's face is not correctly identified as a match.

False positive: A result indicating that a particular condition or characteristic is present when it is absent, typically in the context of facial recognition, referring to instances where a person's face is incorrectly identified as a match.

Haar Cascade Algorithm: A machine learning-based approach used for object detection, particularly in recognizing faces in images or videos by analyzing features at different scales.

Haar Cascade Classifier: A trained model based on the Haar Cascade Algorithm, specifically designed to detect objects such as faces by analyzing patterns of contrast in images.

Local Binary Patterns Histograms (LBPH) is a texture descriptor commonly used in face recognition that analyzes local texture by comparing each pixel to its neighboring pixels, creating binary codes based on their relative intensities.

minNeighbor: A parameter used in object detection algorithms like Haar Cascade, specifying how many neighbors each candidate rectangle should have to retain it.

minSize: A parameter used in object detection algorithms like Haar Cascade, specifying the minimum possible object size for detection.

scaleFactor: A parameter used in object detection algorithms like Haar Cascade, specifying how much the image size is reduced at each image scale.

True negative: A result indicating that a particular condition or characteristic is absent and correctly identified as such. In facial recognition, this refers to instances where a person's face is correctly identified as not matching any known individuals

True positive: A result indicating that a particular condition or characteristic is present and correctly identified as such. In facial recognition, this refers to instances where a person's face is correctly identified as a match to a known individual.

Chapter Two

REVIEW OF RELATED LITERATURE

2.1 Related Literatures

Foreign Literatures

Haar Cascade Algorithm, developed by Paul Viola and Michael Jones, are well-known for their significant contribution to object detection. Their innovative approach has greatly influenced subsequent research in the field, including the work described in the paper “Rapid object detection using a boosted cascade of simple features”. By introducing efficient methods for feature computation, such as the "integral image" representation, and utilizing learning algorithms like AdaBoost (Viola & Jones, 2001).

Viola and Jones' cascade method efficiently focuses on important areas while swiftly ignoring background noise, making it essential for modern object detection. Thus, the legacy of the Haar Cascade classifiers resonates strongly with the advancements presented in the research paper, underscoring their enduring impact on the field of computer vision.

These classifiers excel in identifying facial features, thus forming the foundation for various applications ranging from security systems to image processing algorithms. In recent years, there has been a notable increase in the integration of Haar Cascade classifiers into real-world systems, accompanied by ongoing exploration by researchers into novel methods for maximizing their utility and effectiveness.

According to the study “Use of Haar Cascade Classifier for Face Tracking System in Real Time Video.” (Wanjale et al., 2013) introduced a comprehensive face detection and tracking system designed for real-time video inputs, emphasizing its application in security contexts. Incorporating the Haar-Cascade method and OpenCV libraries, the system's initial phase encompasses face recognition and detection. Subsequently, face tracking is performed using a Face clustering algorithm. The system primarily targets security purposes, operating on video recordings from public areas to identify and track individuals or suspicious activities.

The paper “A Novel Real-Time Face Detection System Using Modified Affine Transformation and Haar Cascades” addresses the challenges posed by tilted, occlusion, and varying illumination in computer vision (Sharma et al. 2018). Their approach employed a Haar-cascade classifier enhanced with Modified Census Transform features, traditionally inadequate for detecting faces under such conditions.

In the study “Analyzing Of Different Features Using Haar Cascade Classifier” conducted by (Yustiawati et al., 2018) discusses the utilization of face recognition for enhancing security measures focusing on the application of a Haar cascade classifier. Their research aims to investigate the effectiveness of using the Haar cascade classifier for analyzing different features, particularly in the context of identifying and matching faces for access control purposes. Through experimental testing, the study demonstrates the efficacy of the proposed method in accurately processing the features of objects, particularly in the context of facial recognition for room security applications.

Another study named “Multi-Faces Recognition Process Using Haar Cascades and Eigenface Methods” reveals how traditional face recognition suffers from processing time and accuracy (Mantoro et al., 2018). To overcome these problems, the study proposes an accelerated approach. By combining Haar Cascades and Eigenface methods, they achieved remarkable results, detecting multiple faces with 91.67% accuracy. By Utilizing OpenCV, specifically designed for Haar-cascade classifiers, the system meticulously examines images, categorizing them as "face" or "not face." Operating within a fixed scale, the classifier continuously analyzes until a face is detected, using data from an XML file for classification decisions.

In “Face Detection Using Haar Cascade in Difference Illumination” (Hapsari et al., 2018) conducted a study exploring the dynamic nature of facial features and their importance across various applications in computer image processing. The research highlighted the challenges in face detection due to the dynamic characteristics of facial features, emphasizing the pivotal role of object detection techniques, particularly in identifying faces within images or video frames. Haar cascade features were specifically noted for their effectiveness in enriching simple features and their efficient processing capabilities. The findings underscored the promising potential of Haar cascade features, particularly in addressing face detection challenges posed by varying illumination conditions.

In their study (Saini et al., 2019) presents an approach for facial recognition aimed at achieving real-time recognition within high-resolution videos using Haar feature-based cascade classifiers. This method is particularly relevant in security applications and attendance systems, offering an alternative to fingerprint-based systems. Leveraging the OpenCV library, the methodology operates in near real-time with a low false alarm rate. Test results across various scenarios, including different obstacles, illuminations, and facial orientations, demonstrate the robustness and superior performance of the proposed approach. Notably, its simplicity and ease of implementation contribute to its effectiveness.

“Face detection using Haar cascade classifier” proposed by (Pandey et al., 2019) explores the efficacy of the Haar cascade classifier as one of the best techniques for object detection, including face detection. Leveraging predefined datasets stored in XML format, the classifier operates by converting real-time face RGB colors into grayscale, facilitating comparison with the dataset to detect faces accurately. This workflow underscores the effectiveness of the Haar cascade classifier in the domain of face detection.

(Gangopadhyay et al., 2019) proposed a process for face detection and expression recognition, presenting a comprehensive approach to recognizing eight expressions: happy, angry, surprise, contempt, disgust, fear, surprise, and neutral. The face detection process utilizes the Haar classifier, known for its robustness and accuracy in detecting facial features.

In the paper “Automatic Face Recognition and Detection Using OpenCV, Haar Cascade and Recognizer for Frontal Face “(Arya & Tiwari, 2020) investigated real-time facial recognition across varying angles and lighting conditions. Employing recognizers such as Eigenface, Fisherface, and LBPH in conjunction with Haar cascade, the algorithms were initially trained with a database of images before testing with real-time captures.

The study "Face Recognition based Attendance System using Haar Cascade and Local Binary Pattern Histogram Algorithm" They developed a robust attendance system based on face recognition to track student attendance accurately (Chinimilli et al., 2020). In their study, they addressed some of the false positives by incorporating a strong threshold based on Euclidean distance values during the detection of unknown individuals. Comparative analysis revealed the superiority of the Haar cascade algorithm with Local Binary Pattern Histogram (LBPH) algorithm over other Euclidean distance-based algorithms like Eigenfaces and Fisherfaces. The Haar cascade was chosen for face detection due to its robustness, while the LBPH algorithm was employed for face recognition, renowned for its robustness against monotonic grayscale transformations. The system achieved a student face recognition rate of 77% with a false positive rate of 28%. Even in scenarios where students wore glasses or had facial hair, the system exhibited remarkable performance. Additionally, the recognition rate for unknown individuals reached nearly 60%, with false positive rates of 14% and 30% with and without applying a threshold, respectively. This highlights the significance of Haar cascade in developing efficient face recognition-based attendance systems.

(Ahmad et al., 2021) Examined the capabilities of Haar Cascades classifier for Real-time face recognition of video surveillance systems in their study. This project integrates face recognition into CCTV systems for real-time detection and identification. The proposed system utilizes a Haar cascade classifier for automatic identification of individuals. Hardware components include a Raspberry Pi processor and Pi Camera module. Development phases involved data gathering, training the recognizer, and face recognition processes, all implemented using Python programming and the OpenCV library on a Raspbian operating system. The system successfully recognizes human faces within specific parameters, such as facial angle, lighting conditions, and distance. This innovative approach aims to reduce the need for manual identification, thus enhancing real-time identification capabilities in surveillance scenarios.

Then (Safiullina et al., 2021) conducted a comprehensive evaluation of Haar cascade classifiers for face recognition in biometric systems, aiming to identify the most suitable classifier based on performance and accuracy metrics. The study assessed four types of Haar cascade classifiers available in the OpenCV library: haarcascade_frontalface_default, haarcascade_frontalface_alt, haarcascade_frontalface_alt2, and haarcascade_frontalface_alt_tree. Through experiments conducted within a developed framework, the training and recognition times, as well as the False Acceptance Rate (FAR) and False Rejection Rate (FRR), were considered for each classifier. The evaluation utilized a dataset comprising 1,120 images collected from 150 users. Results indicated that haarcascade_frontalface_alt and haarcascade_frontalface_alt2 classifiers demonstrated superior performance and were recommended for addressing face recognition challenges in biometric systems.

The research paper "Optimal Face Recognition System using Haar Classifier" address the challenges factors of lighting, background noise, and image quality by evaluating different face recognition algorithms, including Haar classifiers, default face recognizer, Local Binary Pattern Histogram (LBPH) face recognizers, Adaboost algorithm, and Eigenface methods (Wattamwar et al., 2021). Focusing on the utilization of Haar cascades and the LBPH Face recognizer algorithm, the study aims to develop a hybrid approach for a smart voting system authentication. By identifying the presence of faces in images, determining the number of faces, and assessing confidence values, the research aims to propose an optimal algorithm for face detection and recognition, contributing to the advancement of biometric systems.

In their study, (Choi et al., 2022) introduced a pre-processing method for enhancing the efficiency and accuracy of face detection using Haar cascade classifiers. Recognizing the significance of face detection systems in security and human management fields, the study aimed to address the limitations of traditional Haar cascade classifiers, particularly in terms of processing time and false positives. By incorporating a 2D Haar discrete wavelet transform-based pre-processing technique, the proposed method achieved notable improvements in processing speed and false positive reduction. Experimental results using both public and private datasets demonstrated a significant enhancement in processing speed by 32.05% and a reduction in false positives by 25.46% compared to conventional histogram equalization methods. Moreover, the proposed method exhibited similar performance to image contraction-based approaches while outperforming Gaussian filter-based methods in terms of false positive reduction without compromising true positive detection rates. These findings highlight the effectiveness of Haar cascade classifiers enhanced by vertical component calibration for robust and efficient face detection applications.

The proposed study by (Diyasa et al., 2022) on “Feature Extraction for Face Recognition Using Haar Cascade Classifier” discusses the utilization of the Haar Cascade Classifier method in facial extraction, supplemented by CNN for facial classification, within the context of developing facial recognition systems. They recognize the pivotal role of accurate recognition, which can be influenced by factors such as lighting conditions, facial expressions, positions, and attribute changes. Employing Python programming and the OpenCV library, this research underscores the efficacy of Haar Cascade Classifier in feature extraction for enhancing face recognition systems.

According to the study "Human Face Recognition Applying Haar Cascade Classifier”, Human face recognition relies on two stages which is identification and recognition. While typically performed using full-frontal faces, challenges arise when dealing with partial facial data, such as that obtained from CCTV cameras, which remains an underexplored area in computer-based face recognition research (Shamrat et al., 2022). They propose a comprehensive framework for face recognition using the haar cascade classifier, comprising three key phases: Face Data Gathering (FDG), Train the Stored Image (TSI), and Face Recognition using Local Binary Patterns Histograms (LBPH) algorithm. Through experimentation and analysis, the researchers demonstrate the effectiveness of Haar feature selection, integral image generation, Adaboost preparation, and cascading classifiers in enhancing facial feature detection and classification accuracy. Moreover, they integrate LBPH estimation to further refine the recognition framework, leveraging a dataset and algorithmic parameters to optimize performance. By employing a combination of LBPH operation and histogram extraction. The study results in a strong computational framework for human face recognition, highlighting the effectiveness and versatility of haar cascade classifiers.

Building upon the facial recognition capabilities of Haar Cascade classifiers, (Bairagi et al., 2022) established the foundation for their Real-time Face Recognition Smart Attendance System Using Haar Cascade Classifiers in student attendance tracking. These classifiers were instrumental in their object detection framework, enabling the accurate identification of facial features within images. Given the difficulties associated with manual tracking attendance procedures, researchers found biometric alternatives such as fingerprints and QR codes. However, as face recognition technology has emerged, the process has become even more streamlined. Using the capabilities of machine learning, particularly Haar Cascade classifiers, the study explores the feasibility of implementing facial recognition for attendance management.

The study presents a smart attendance system that employs machine learning-based techniques, including Haar Cascade classifiers for object detection and LBPH (Local Binary Patterns Histograms) for face recognition. Through experimentation and evaluation, the proposed system demonstrates promising results in accurately detecting and recognizing faces, offering a more efficient alternative to traditional attendance-taking methods.

The study “Face detection by using Haar Cascade Classifier” (Mccullagh, et. al., 2023) Analyze the extensive application of the Haar Cascade Classifier for face detection, highlighting its capability and efficiency in identifying objects in images and videos. The algorithm's multi-stage process involves trained datasets, extracting features, and building cascade classifiers to detect faces accurately. Despite its proven accuracy and efficiency, the algorithm faces challenges under adverse conditions, such as challenging lighting or partial occlusion. Nevertheless, it remains a widely used tool in various fields, including facial recognition, security, and computer vision research. Researchers continue to explore more advanced techniques, such as the Histogram of Oriented Gradients (HOG) and Deep Neural Network (DNN), to address the limitations of the Haar Cascade Classifier and enhance its performance further. Thus, the Haar Cascade Classifier stands as a powerful tool for face detection, with considerations for its limitations and the potential for more advanced alternatives in specific applications.

According to the study “Localizing Face Recognition with Haar-Cascade Classifier and LBPH using Python” (Legaspi, 2023) employ the integration of the Haar-Cascade Classifier and Local Binary Pattern Histogram (LBPH) in developing a Face Recognition System using Python. With a dataset consisting of 1000 photos per individual gathered through Python scripting, the model underwent training, identification, and recognition processes. Achieving an overall efficiency rating of 84%, this study presents a practical recommendation for the utilization of the combined approach. The findings contribute as a valuable reference for further advancements in face recognition systems, particularly in conjunction with other image classification algorithms

Local Literatures

In 2011, (Orquez et al., 2011) conducted a study titled "Enhancement of Windows User Access Security Using Multi-Feature Face Detection and Recognition" aimed at improving user access security on Windows systems through the development of a multi-feature face detection and recognition system. This study specifically focused on laptop brands such as Asus, Lenovo, and Toshiba, which offer built-in face recognition features. The researchers utilized OpenCV's Haar Cascade for face detection and Eigenface Recognizer for face recognition. Additionally, they implemented a blink detection algorithm to enhance security by ensuring that the recognized face is of a live person and not a photograph. The system was tested biometrically to evaluate its performance and accuracy.

The study concluded that incorporating blink detection into face recognition systems significantly enhances security. This method effectively counters attempt to bypass authentication and access control using photos of authorized users. The findings suggest that multi-feature recognition systems, which combine face detection with blink detection, can provide a more robust security solution for Windows user access.

(Ramos et al., 2019) conducted a study to evaluate the impact of various lighting conditions, angles, and distances on the performance of a face recognition system used for door access control. The system utilized a Raspberry Pi controller and incorporated the Haar-Cascade algorithm for face detection, Principal Component Analysis (PCA) for feature extraction, Support Vector Machine (SVM) for classification, and Euclidean Distance for recognition. The researchers created 240 training datasets and tested the model with five subjects. The recognition performance was 100% under fluorescent light, 74.65% under candlelight, and 88% under flashlight, resulting in an average accuracy of 89.5%. The study found that the system performed best when the subject faced the camera directly, achieving a 100% recognition rate. However, recognition accuracy decreased to 82.3% when the subject faced left or right or was 1-3 meters away. Similarly, upward or downward angles and the same distance range resulted in an 88% accuracy rate. The overall accuracy of the model across all experiments was 89.5%.

In the same year of 2019, (Ramos et al., 2019) conducted a study titled "Filipino-Based Facial Emotion Features Datasets Using Haar Cascade Classifier and Fisherfaces Linear Discriminant Analysis Algorithm." The research aimed to apply Filipino-based facial emotion features by revalidating the available features in the Visage Cloud API. This served as a basis for determining how emotions differ from expert validation and testing through the WEKA tool. The primary focus of their research was to validate the classification accuracy performance of the Fisherfaces Linear Discriminant Analysis (LDA) algorithm. The study achieved a classification accuracy of 90.66% based on the API outcomes with 150 instances and an 83.61% classification rate for 609 features, outperforming existing studies in the field.

According to (Garcia et al.,2020), the use of technology, particularly the reliance on smartphones, has led to the generation and storage of massive amounts of image and video data. Their study focuses on utilizing these images for emotion recognition through deep learning algorithms. The primary goal is to identify emotions in facial images using Convolutional Neural Networks (CNN) and Haar cascades within the OpenCV framework, implemented in Spyder-Python. The study classifies six emotions—happy, sad, angry, surprised, fear, and neutral—among the faculty of the College of Engineering and

Computer Studies (COECS). The emotion recognition system (FER) is intended to analyze instructors' behavioral patterns and their impact on students' learning abilities. The researchers successfully implemented FER, tested it manually, and achieved real-time emotion identification, meeting their study's objectives.

The study by (Yong et al.2023) details the development and evaluation of the RAMY Greeting Feature, a self-maneuvering robot installed in the Asia Pacific College (APC) lobby to act as an information hub. Using HAAR Cascade for facial detection, HOG for feature extraction and classification, and pyttsx3 for text-to-speech greetings, the system detects and recognizes faces, greeting individuals based on their recognition status. The system achieved an accuracy of 85.16%, reliability of 87%, and robustness with a 3.52-meter average detection distance, though performance varied with lighting conditions and image quality. Despite some limitations, the RAMY Greeting Feature effectively detects, recognizes, and greets faces, ensuring privacy and minimal budget requirements.

2.2 Synthesis

Haar Cascade classifiers stand as fundamental pillars in the realm of object detection and facial recognition, showcasing remarkable efficiency across diverse applications ranging from attendance systems to real-time surveillance. These classifiers are renowned for their ability to accurately detect objects and facial features even in complex scenarios, such as varying lighting conditions or partial occlusion.

Python libraries, particularly OpenCV and face_recognition, play a significant role in augmenting the capabilities of Haar Cascade classifiers. By leveraging these libraries, developers can harness the power of Haar Cascade classifiers within their Python-based projects, ensuring robust performance and streamlined implementation.

While alternatives to Haar Cascade classifiers emerge, their enduring relevance and widespread adoption underscore their significance in shaping the future of computer vision. Python, with its versatile libraries and frameworks, serves as a cornerstone in harnessing the capabilities of Haar Cascade classifiers for a myriad of applications. As research and development progress, the synergy between Haar Cascade classifiers and Python-based tools continues to drive innovation, paving the way for even more sophisticated applications in the field of computer vision.

Chapter Three

DESIGN AND METHODOLOGY

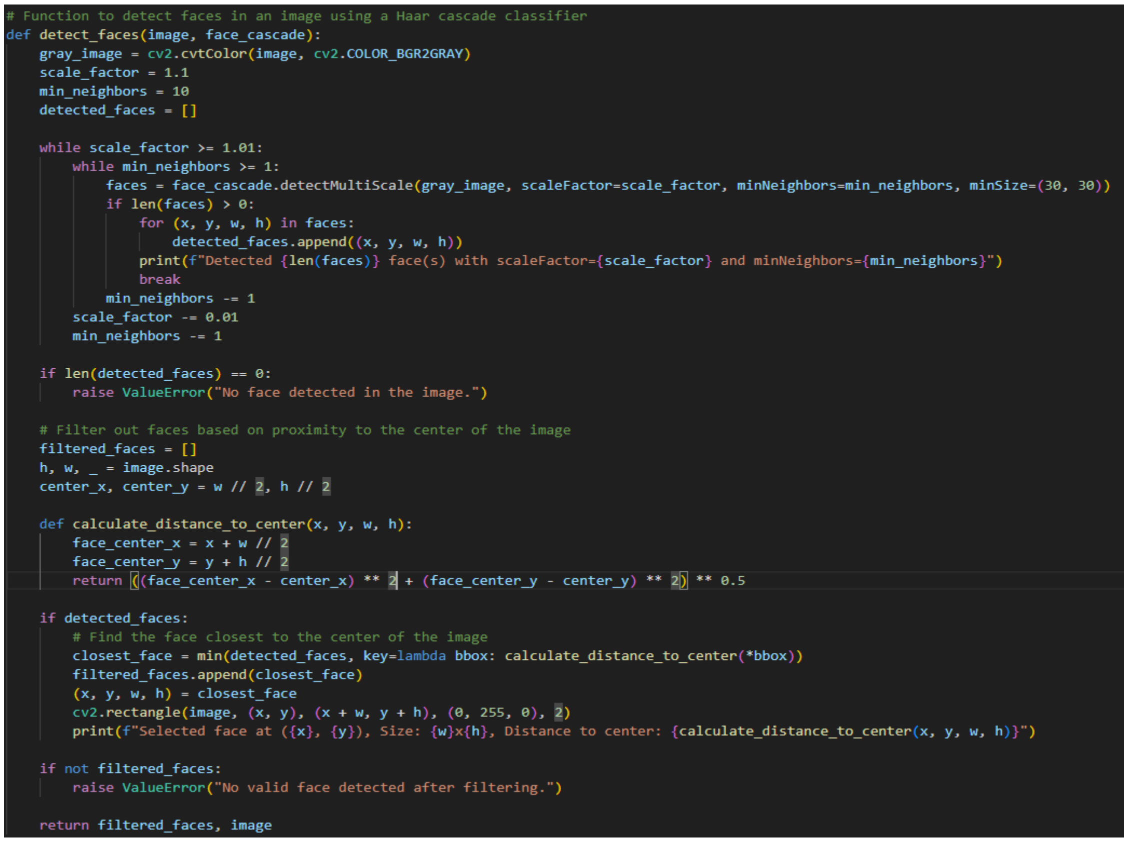
In this chapter, the researcher will outline the design and methodology of the study, detailing both the framework and approach used to achieve the research objectives. The design includes an explanation of the Enhanced Haar Cascade Algorithm, which is used for face detection, and the system architecture, illustrating how the system will be structured and developed. The methodology covers the data gathering procedure for the dataset, the analysis of this data, and the solutions proposed to address the identified problems. Together, these elements provide a comprehensive guide to the structure and execution of the research process.

3.1 Enhanced Haar Cascade Algorithm Pseudocode

```
STEP 1:
        1.1 Load images using OpenCV's imread function.
STEP 2:
        2.1 Convert image to grayscale.
        2.2 Detect faces using Haar cascade classifier in grayscale images. (scaleFactor,
        minNeighbors, and minSize are used for face detection.)
        2.3 Initialize scaleFactor to 1.1 and minNeighbors to 10, then loop until
        scaleFactor = 1.01 and minNeighbors = 1.
        2.4 Select the largest detected face that is close to the center of the image.
        2.5 Put boxes in the detected faces
STEP 3:
        3.1 Convert the detected faces from grayscale to RGB
        3.2 Extract the face encodings of each face using face_recognition and
        face_encoding
STEP 4:
        4.1 Compute the Euclidean distance between the two extracted encodings.
        4.2 Convert the distance to a similarity percentage
STEP 5:
        5.1 Show images and their similarity percentage
```

3.2 Enhanced Haar Cascade Algorithm

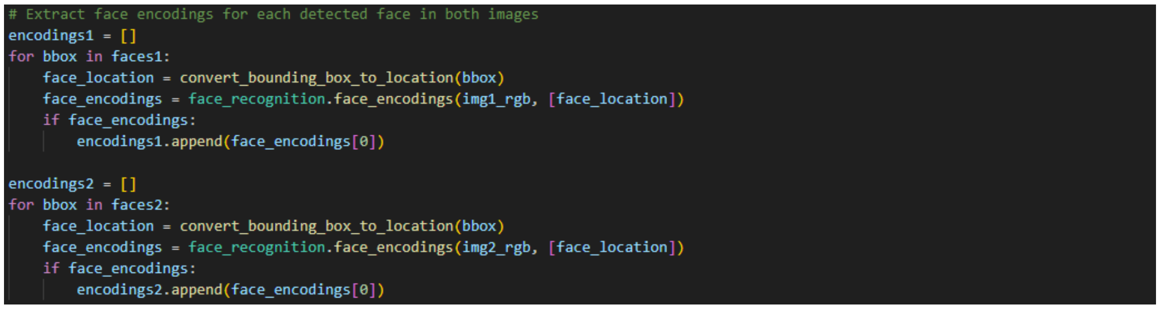
The Enhanced Haar Cascade Algorithm combines the strengths of the Haar Cascade Algorithm and the face_recognition library to perform a comprehensive analysis of facial features in two distinct images. This process is initiated by the ingestion of the images, followed by a conversion to the RGB format necessary for the operations of the face_recognition library.



```python
# Function to detect faces in an image using a Haar cascade classifier
def detect_faces(image, face_cascade):
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    scale_factor = 1.1
    min_neighbors = 10
    detected_faces = []

    while scale_factor >= 1.01:
        while min_neighbors >= 1:
            faces = face_cascade.detectMultiScale(gray_image, scaleFactor=scale_factor, minNeighbors=min_neighbors, minSize=(30, 30))
            if len(faces) > 0:
                for (x, y, w, h) in faces:
                    detected_faces.append((x, y, w, h))
                print(f"Detected {len(faces)} face(s) with scaleFactor={scale_factor} and minNeighbors={min_neighbors}")
                break
            min_neighbors -= 1
        scale_factor -= 0.01
        min_neighbors -= 1

    if len(detected_faces) == 0:
        raise ValueError("No face detected in the image.")

    # Filter out faces based on proximity to the center of the image
    filtered_faces = []
    h, w, _ = image.shape
    center_x, center_y = w // 2, h // 2

    def calculate_distance_to_center(x, y, w, h):
        face_center_x = x + w // 2
        face_center_y = y + h // 2
        return ((face_center_x - center_x) ** 2 + (face_center_y - center_y) ** 2) ** 0.5

    if detected_faces:
        # Find the face closest to the center of the image
        closest_face = min(detected_faces, key=lambda bbox: calculate_distance_to_center(*bbox))
        filtered_faces.append(closest_face)
        (x, y, w, h) = closest_face
        cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
        print(f"Selected face at ({x}, {y}), Size: {w}x{h}, Distance to center: {calculate_distance_to_center(x, y, w, h)}")

    if not filtered_faces:
        raise ValueError("No valid face detected after filtering.")

    return filtered_faces, image
```

Figure 11 Enhanced Haar Cascade Parameters

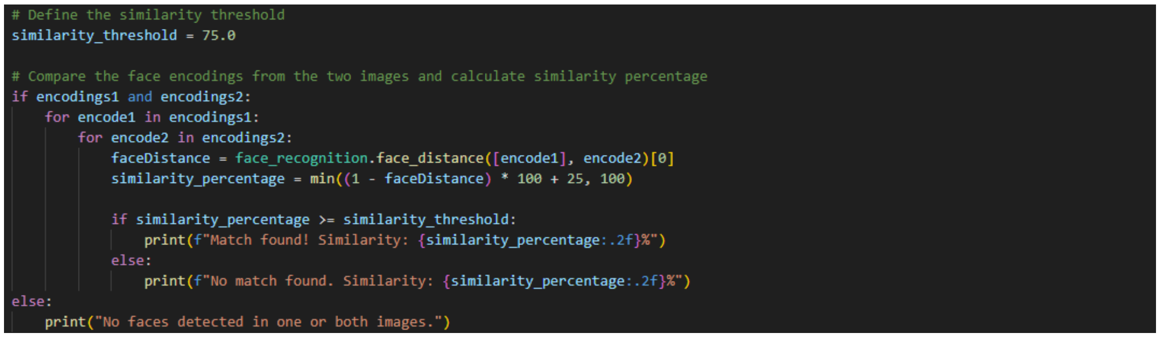
Figure 11 shows the parameters of the Enhanced Haar Cascade Algorithm. The core of this process uses the Haar Cascade classifier, a machine learning tool for detecting faces. It looks for facial features like eyes, nose, and mouth by recognizing their spatial arrangement. The classifier works on grayscale images and uses the detectMultiScale function to find possible faces. Instead of manually adjusting the parameters (scaleFactor, minNeighbors, minSize) for each image, we automated the process. The logic iterates through different parameter values and selects the best setting based on filtering. Since faces are usually larger than non-face objects, we choose the largest detected face that is closest to the center of the image as the real face.



```
# Extract face encodings for each detected face in both images
encodings1 = []
for bbox in faces1:
    face_location = convert_bounding_box_to_location(bbox)
    face_encodings = face_recognition.face_encodings(img1_rgb, [face_location])
    if face_encodings:
        encodings1.append(face_encodings[0])

encodings2 = []
for bbox in faces2:
    face_location = convert_bounding_box_to_location(bbox)
    face_encodings = face_recognition.face_encodings(img2_rgb, [face_location])
    if face_encodings:
        encodings2.append(face_encodings[0])
```

Figure 12 Enhanced Haar Cascade with face_recognition

Figure 12 shows the face encoding process of the Enhanced Algorithm. Once the facial regions are successfully identified, the methodology proceeds with converting these grayscale regions back to RGB format, which is necessary for the face_recognition library to function effectively. This library utilizes advanced deep learning models to generate numerical encodings that encapsulate the unique features of the detected faces. These encodings from face_encoding, derived from both images, are then subjected to a comparative analysis to determine the degree of similarity between the faces.



```
# Define the similarity threshold
similarity_threshold = 75.0

# Compare the face encodings from the two images and calculate similarity percentage
if encodings1 and encodings2:
    for encode1 in encodings1:
        for encode2 in encodings2:
            faceDistance = face_recognition.face_distance([encode1], encode2)[0]
            similarity_percentage = min((1 - faceDistance) * 100 + 25, 100)

            if similarity_percentage >= similarity_threshold:
                print(f"Match found! Similarity: {similarity_percentage:.2f}%")
            else:
                print(f"No match found. Similarity: {similarity_percentage:.2f}%")
else:
    print("No faces detected in one or both images.")
```

Figure 13 Enhanced Haar Cascade Similarity calculation

Figure 13 shows the similarity rate calculation. The comparative analysis hinges on calculating a similarity percentage based on the distance between the facial encodings. A critical element in this process is the establishment of a similarity threshold. This threshold, which can be adjusted to meet specific requirements, plays a pivotal role in classifying pairs of faces as either matching or non-matching. The flexibility of this threshold allows for the modulation of the sensitivity of the facial matching process, making it adaptable to different application contexts. In essence, this process synergistically combines the Haar Cascade's facial detection capabilities with the deep learning-driven encoding and comparison functions of the face_recognition library.

3.3 System Architecture

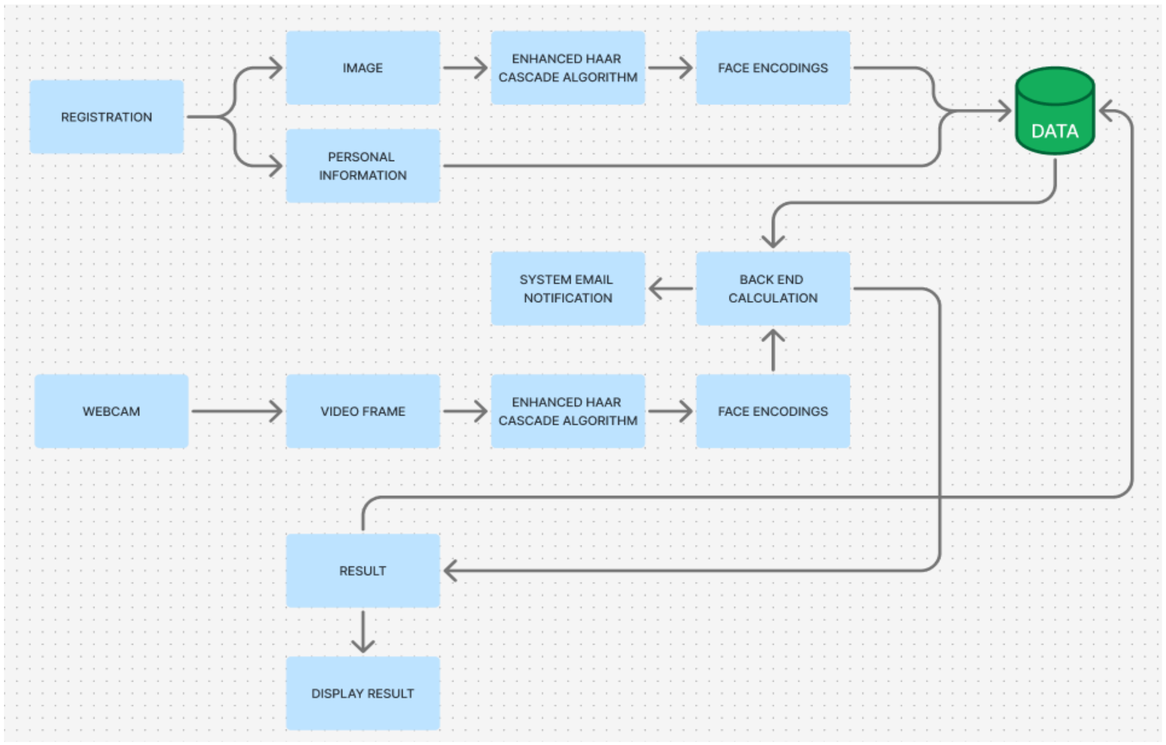


Figure 14 System Architecture of the Face Recognition System with Enhanced Haar Cascade Algorithm

Figure 14 illustrates the system architecture of the facial recognition system, which utilizes the enhanced Haar Cascade algorithm. The process begins at the registration stage, where each person’s image and personal information are collected and submitted. Each image is processed by the enhanced Haar Cascade algorithm to generate encodings, which are then stored in the database along with the individual's information.

In real-time recognition, the system captures live video from a webcam. Each video frame is processed by the Haar Cascade algorithm, generating an encoding that is compared to the stored encodings in the database. If a match is found, the system displays the person as "recognized" alongside their stored information. If no match is found, the individual is classified as "unknown," and the system sends an alert notification to the appropriate authority. This process ensures real-time identification and alerts for unregistered individuals.

3.4 Data gathering procedures and Data analysis method

The researchers will employ a systematic approach to collect a comprehensive dataset of facial images, ensuring a diverse range of conditions and characteristics to robustly test the enhanced facial recognition algorithm and compare it side by side with the Haar Cascade Algorithm with LBPH. The procedure is as follows:

1. **Selection of images:**
   A total of 263 well-known individuals will be chosen to provide facial images. These participants will represent a diverse range of facial features and demographics, including various ages, genders, and ethnicities. Additionally, 12 non-face images will be included.
2. **Environmental Variations:**
   To ensure the dataset encompasses various real-world scenarios, images will be captured under different environmental conditions. This includes settings with complex backgrounds, dynamic lighting (such as varying light intensities and angles), and additional elements like glasses and varying hairstyles.
3. **Image Collection:**
   Each selected famous personalities and non-face images will have a pair of different images taken from the internet. (Google, Facebook, Instagram, etc.) resulting in a total of 550 images. The dual-image approach per individual aims to capture slight variations in expression, angle, or lighting to test the algorithm's robustness and consistency.

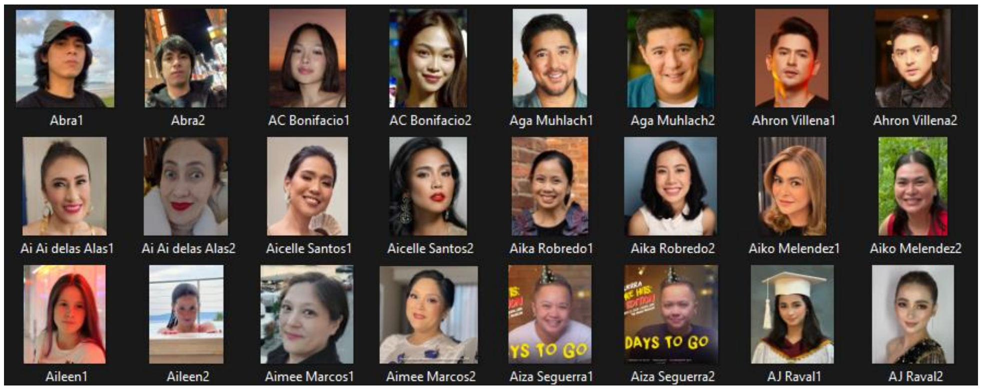


Figure 15 Data set for testing

Figure 15 shows the sample image dataset for testing, both the enhanced and existing algorithm will use the given dataset to have pair competition and accurate interpretation for the result.

The researchers will carry out a comprehensive testing phase to evaluate the performance of the enhanced Haar Cascade Algorithm and the integrated face recognition model. The primary metrics for evaluation will include Confusion Matrix, which has Accuracy, Precision, Recall, and F1 Score. The testing process will be automated to ensure efficiency and reproducibility. The detailed procedures are as follows:

Automated Testing Process:

- An automated testing script will be developed to handle the pairwise comparison of images. This script will ensure that each image is compared to every other image in the dataset.
- The script will iterate through the dataset, comparing each image (denoted as <image name>1 or <image name>2) to all other images except for the same file name. This will be repeated for every image in the dataset.
- A total of 301,950 comparison will be made, considering the 549 x 550 computation.

To interpret the data from the confusion matrix, we will solve it for Accuracy, Precision, Recall and F1 Score.

- **Accuracy:** The proportion of true results (both true positives and true negatives) among the total number of cases examined.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} X\ 100$$

- **Precision:** The proportion of true positive results among all positive results predicted

$$Precision = \frac{TP}{TP + FP} x\ 100$$

- **Recall:** The proportion of true positive results among all actual positive cases.

$$Recall = \frac{TP}{TP + FN} x\ 100$$

- **F1 Score**: The harmonic mean of precision and recall, providing a single metric that balances both concerns.

$$F1\ Score = \frac{2\ x\ Precision\ x\ Recall}{Precision + Recall}$$

3.5 Solutions to the Problems

To address the inaccuracy of the Haar Cascade Algorithm particularly in scenarios with variations in lighting, facial expressions, and occlusions that results in False Positive or False Negative. The researchers propose the modification in Haar Cascade Algorithm by implementing open-source face_recognition library. The face_recognition library is a popular Python package used for recognizing, and analyzing faces in images it computes face encodings, which are 128-dimensional vectors representing the unique features of a face. Similarity is based on the Euclidean distance between these feature vectors.

1. Calculate the Euclidean distance of face encoding. Each image has their own unique 128 face encoding

- Given two face encodings:
  - Face encoding 1: $E_1 = [E_{1,1},\ E_{1,2}, \ldots, E_{1,128}]$
  - Face encoding 2: $E_2 = [E_{2,1},\ E_{2,2}, \ldots, E_{2,128}]$

For two face encodings, $E_1$and E_2, the Euclidean distance d_{face} is:

$$d_{face} = \sqrt{\sum_{i=1}^{128} \left(E_{1,i} - E_{2,i}\right)^2}$$

d_{face} = is the distance of the two face encodings

$\sum_{i=1}^{128} \left(E_{1,i} - E_{2,i}\right)^2$ = summation of the 128 face encodings from the images

2. Calculate similarity percentage:

- The similarity percentage can be calculated as:

$$Similarity\ Percentage = \left(1 - \frac{d_{face}}{d_{max}}\right) \times 100 + 25$$

Where:
  - If $d_{face} = 0$, it indicates a perfect match, resulting in 100% similarity
  - If $d_{face} = d_{max}$, it indicates a not match, resulting in 0% similarity
  - $d_{max} = 1.0$, is the assumed maximum distance

With this calculation, we can assume the similarity percentage of the two images, if the similarity percentage is ≥ 75% then the images is taken as match, if not then it's not match

To address the problem with individuals sharing similar attributes (e.g., wearing glasses), leading to potential false positives. The researchers propose to add additional step in face encoding. face_encoding computes the 128-dimensional vectors from RGB images, not grayscale images. The following are the steps to convert the image to RGB while retaining the function of Haar Cascade Algorithm for grayscale image.

1. Haar Cascade Algorithm will convert the image to **grayscale** for face detection.
2. The coordinates of the detected face from the **grayscale** image will be saved.
3. The **grayscale** image will be converted to **RGB.**
4. The **RGB** image will use the saved coordinates where the face is detected.
5. Finally, the coordinates where the face is located will be encoded using the face_encoding where the 128-dimensional vectors are extracted.

To address the problem with detecting real faces, resulting in false negatives and the cost of some wrong acceptance of non-human faces (false positive). The researchers will implement the usage of the Haarcascade pre-trained XML file together with the modification in detectMultiScale such as scaleFactor, minNeighbors, and minSize by transforming it to a logical process and filtering than the standard baseline tweaking. Instead of using the traditional usage of changing the baseline values we will create a logic process where it will iterate thru several parameter values and filter the result.

Let:

S be the initial scaleFactor value (e.g., 1.1)

ΔS be the decrement step for scaleFactor (e.g., 1.01)

M be the initial minNeighbors value (e.g., 10)

ΔM be the decrement step for minNeighbors value (e.g., 1)

N be the number of detected faces.

The number of times the inner loop runs for a given scaleFactor value is $M - 1$ because the minNeighbors decrements from $M\ to\ 1$.

The overall process can be represented by:

$$Total\ Faces\ Detected = \sum_{i=0}^{k-1} \sum_{j=0}^{M-1} N_{i,j}$$

Where:

i iterates over the range of scaleFactor values from S to 1.01, decreasing by ΔS at each step

j iterates over the range of minNeighbors values from M to 1, decreasing by ΔM at each step

$N_{i,j}$ represents the number of faces detected when using scaleFactor equal to $S - i \; x \; \Delta S$ and minNeighbors equal to $M - j \; x \; \Delta M$

In this equation:

The outer summation $\sum_{i=0}^{k-1}$ accounts for different values of scaleFactor and the inner summation $\sum_{j=0}^{M-1}$ accounts for different values of minNeighbors at each scaleFactor

For the filtering process it will be based on proximity to the center of the image, as an equation, we can define it using the following steps:

Calculate the Center of the Image: Let (w, h) be the width and height of the image.

- Center coordinates: (C_x, C_y), where $C_x = \frac{w}{2}$ and $C_y = \frac{h}{2}$

Calculate the Distance to the Center: For each face with bounding box $(x_i, y_i, w_i, y_i,)$

- Face center coordinates: (F_x, F_y), where $F_x = x_i + \frac{w_i}{2}$ and $F_y = y_i + \frac{h_i}{2}$
- Distance to the center: D_i, where

$$D_i = \sqrt{(F_x - C_x)^2 + (F_y - C_y)^2}$$

Or

$$D_i = \sqrt{\left(\left(x_i + \frac{w_i}{2}\right) - C_x\right)^2 + \left(\left(y_i + \frac{h_i}{2}\right) - C_y\right)^2}$$

Therefore, the logical process and face filtering always select the real face and neglects the false positive detection.

Chapter Four

RESULTS AND DISCUSSION

4.1 Haar Cascade similarity measurement vs face_recognition similarity measurement

Applying the solutions from chapter 3, the researchers will conduct testing and comparison between the Haar Cascade Algorithm and Enhanced Haar Cascade Algorithm when it comes to similarity measurement (faces without accessories, under complex background)

Addressing Statement of the Problem no.1

Haar Cascade similarity measurement

- Calculate the Euclidean Distance of the histogram
  - The Euclidean distance between the histogram of two faces, “hist1” and “hist2”, is given by:

$$d(hist1, hist2) = \sqrt{\sum_{i=0}^{P+1} (hist1_i - hist2_i)^2}$$

Where:

- hist1 is the LBP Histogram of the first face
- hist2 is the LBP Histogram of the second face

- Calculate Similarity Percentage:
  - The similarity percentage is calculated using the formula:

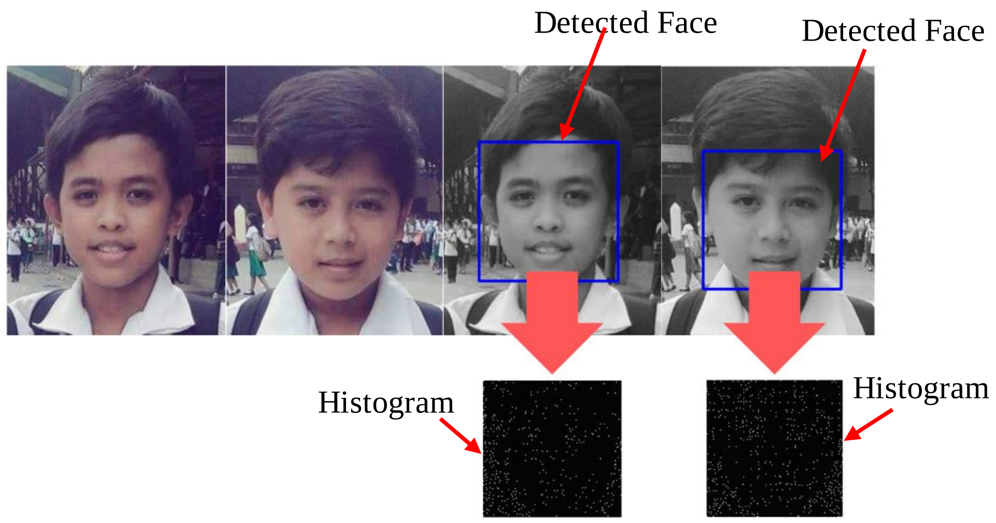
$$Similarity\ Percentage = 100\ x\ \left(1 - d(hist1, hist2)\right)$$

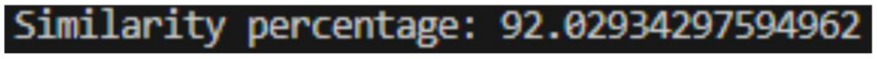
Or

$$Similarity\ Percentage = 100\ x\ \left(1 - \sqrt{\sum_{i=0}^{P+1} (hist1_i - hist2_i)^2}\right)$$

Example:

- Using 2 different images, we will compute for their similarity using the above-mentioned formula.





```
Similarity percentage: 92.02934297594962
```

Figure 16 Face histogram with Haar Cascade Algorithm

Figure 16 shows a similarity percentage of 92.02% between two different images, the computation is simulated below.

The histogram of each image is:

hist1 = [0.01875556, 0.04324444, 0.03568889, 0.13257778, 0.3072, 0.21848889, 0.07804444 0.05, 0.0524, 0.0636]

hist2 = [0.02451647, 0.05473444, 0.03986667, 0.13034516, 0.2398579, 0.20034209, 0.08885575, 0.06561116, 0.08087365, 0.07499671]

Using the formula:

$$d(hist1, hist2) = \sqrt{\sum_{i=0}^{P+1} (hist1_i - hist2_i)^2}$$

1. Subtract $hist1_i$ *to* $hist2_i$:

   $0.01875556 - 0.02451647 = -0.00576091$
   $0.04324444 - 0.05473444 = -0.01149$
   $0.03568889 - 0.03986667 = -0.00417778$
   $0.13257778 - 0.13034516 = 0.00223262$
   $0.3072 - 0.2398579 = 0.0673421$
   $0.21848889 - 0.20034209 = 0.0181468$
   $0.07804444 - 0.08885575 = -0.01081131$
   $0.05 - 0.06561116 = -0.01561116$

$0.0524 - 0.08087365 = -0.02847365$
$0.0636 - 0.07499671 = -0.01139671$

2. Square each difference:
    - $(-0.00576091)^2 = 3.3181668081 \times 10^{-5}$
    - $(-0.01149)^2 = 1.320601 \times 10^{-4}$
    - $(-0.00417778)^2 = 1.7453862284 \times 10^{-5}$
    - $(0.00223262)^2 = 4.9846030644 \times 10^{-6}$
    - $(0.0673421)^2 = 0.00453598363641$
    - $(0.0181468)^2 = 0.00032930727044$
    - $(-0.01081131)^2 = 1.1688446896 \times 10^{-4}$
    - $(-0.01561116)^2 = 2.4369935336 \times 10^{-4}$
    - $(-0.02847365)^2 = 8.1044709225 \times 10^{-4}$
    - $(-0.01139671)^2 = 1.2984464041 \times 10^{-4}$
3. Sum of squared differences:

$3.3181668081 \times 10^{-5} + 1.320601 \times 10^{-4} + 1.7453862284 \times 10^{-5} + 4.9846030644 \times 10^{-6} + 0.00453598363641 + 0.00032930727044 + 1.1688446896 \times 10^{-4} + 2.4369935336 \times 10^{-4} + 8.1044709225 \times 10^{-4} + 1.2984464041 \times 10^{-4} =$ **0.006046325635**

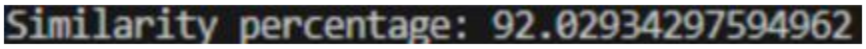
$$\sqrt{0.006046325635} = \mathbf{0.07774498}$$

Then:

Calculate similarity percentage using the formula:

$$Similarity\ Percentage = 100\ x \left(1 - \sqrt{\sum_{i=0}^{P+1} (hist1_i - hist2_i)^2}\right)$$

$$Similarity\ Percentage = 100\ x\ (1 - \mathbf{0.07774498}) \cong \mathbf{92.225502}$$



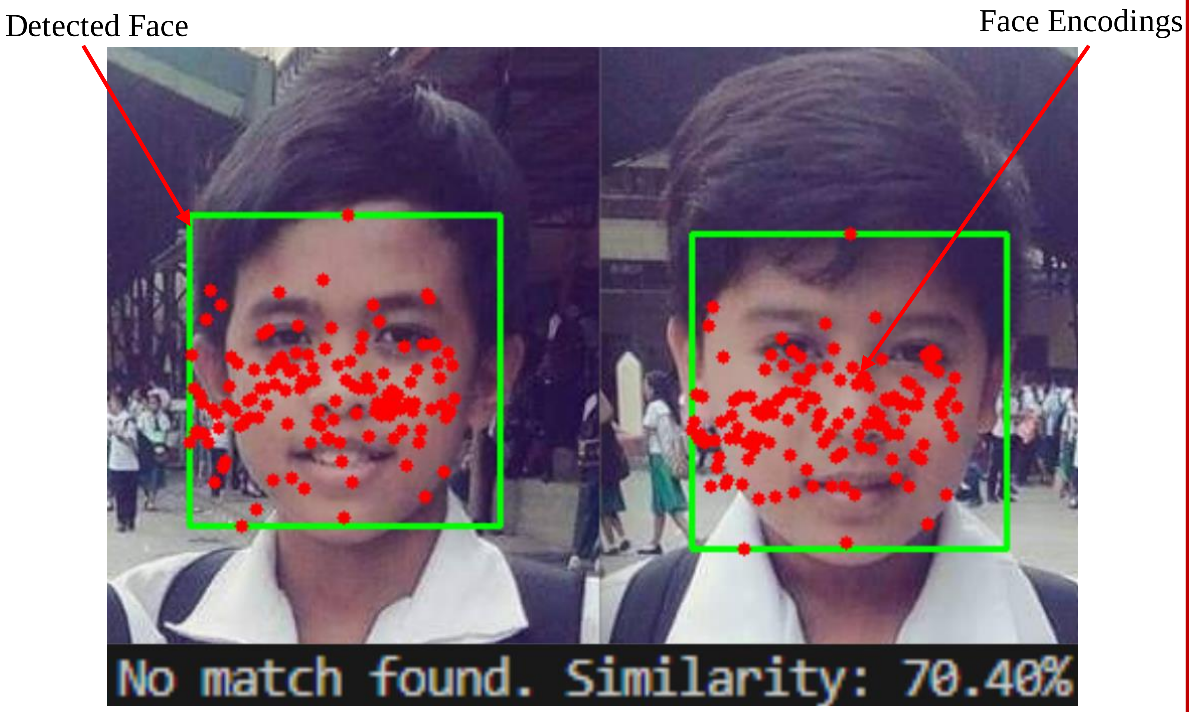
```
Similarity percentage: 92.02934297594962
```

Enhanced Haar Cascade with face_recognition similarity measurement

- The face_recognition library computes face encodings, which are 128-dimensional vectors representing the unique features of a face. Similarity is based on the Euclidean distance between these feature vectors.

Example:

- Using the same image in Figure 3.9, we will calculate the similarity percentage using the formula in chapter 3.



```
No match found. Similarity: 70.40%
```

Figure 17 Face encoding with face_recognition

Figure 17 shows a similarity percentage of 70.40% between two different images after applying the face_recognition library with Haar Cascade Algorithm which turns to be a True Negative, meaning there's a decrease in False Positive rate when it come to face matching. The calculation for the similarity percentage is simulated below.

Using the formula:

$$d_{face} = \sqrt{\sum_{i=1}^{128} \left(E_{1,i} - E_{2,i}\right)^2}$$

Where:

- Face encoding 1: -4.30667996e-02, -2.28661392e-02, 4.41910885e-02, -6.76112324e-02, -5.32401949e-02, 3.75085622e-02, -7.42348135e-02, -7.37156942e-02, 2.18665019e-01, -1.89747572e-01, 2.68395543e-01, -7.25680143e-02, -1.42478496e-01, -1.12361789e-01, -2.46396046e-02, 1.41891658e-01,-1.84912816e-01, -1.71008646e-01, -6.59527630e-02, -5.33263851e-03, 2.74646841e-02, -6.59616068e-02, -3.90837304e-02, 4.00942303e-02,-1.83258519e-01, -3.48887980e-01, -7.70650283e-02, -1.20775163e-01,4.01194692e-02, -6.10612072e-02, -9.04657319e-02, 4.35426505e-03,-2.21657246e-01, -6.42056763e-02, -7.62591977e-03, 1.07554428e-01,1.97649579e-02, -7.71873742e-02, 1.43373460e-01, -4.05969378e-03,-2.16242820e-01, 2.99163368e-02, 3.22665758e-02, 1.87850237e-01,1.16558358e-01, 5.08937463e-02, -2.06838194e-02, -1.09716654e-01,1.55543283e-01, -2.01817676e-01, 4.80858758e-02, 8.97232741e-02, 1.38065040e-01, 1.26853017e-02, 8.10191259e-02, -1.45155400e-01, 3.71416435e-02, 1.10719591e-01, -1.88650250e-01, 3.39751318e-02, -3.29232775e-02, -7.75503740e-03, -3.48072499e-03, -7.82910287e-02, 2.22300291e-01, 1.15258060e-01, -5.40582910e-02, -1.89038217e-01, 1.61642343e-01, -1.14784800e-01, -3.12405005e-02, 8.30037594e-02, -1.05353147e-01, -1.85976774e-01, -

3.30560535e-01, -3.50751914e-03, 4.49433416e-01, 1.13463685e-01, -2.09388241e-01, 7.30885193e-02, -5.45312613e-02, -8.96774530e-02, 9.41628218e-02, 8.74642506e-02, -2.75933668e-02, 6.62853867e-02, -9.01825279e-02, 2.60520726e-04, 2.39508584e-01, -3.01398225e-02, -7.98423029e-03, 1.35592207e-01, -3.17125916e-02, 3.31605934e-02, 2.73950733e-02, -5.68273440e-02, -1.16621196e-01, 3.77129987e-02, -1.67505071e-01, -5.78765646e-02, 1.83115229e-02, 2.59113517e-02, -2.09533591e-02, 7.66970366e-02, -1.88385516e-01, 7.13074952e-02, 2.10754331e-02, -1.06274575e-01, 1.80404559e-02, 4.86092903e-02, -1.18554652e-01, -7.99826458e-02, 1.46843702e-01, -2.83360094e-01, 1.20719150e-01, 1.56977221e-01, 1.52106941e-01, 1.47779733e-01, 1.04518764e-01, 1.99433714e-02, -1.71920471e-03, 3.12098339e-02, -2.08410040e-01, -3.23691256e-02, 4.78609428e-02, -6.29274026e-02, 8.91835839e-02, 1.40212625e-02

- Face encoding 2: -1.25418440e-01, 8.47699940e-02, 1.29728168e-02, -8.97089690e-02,-1.01323418e-01, -6.66908454e-03, -2.99557410e-02, -9.84393209e-02, 1.72883004e-01, -1.20268323e-01, 2.50450104e-01, -4.15269881e-02, -2.16269329e-01, -5.26710786e-02, -8.47432911e-02, 2.03614399e-01, -1.84184834e-01, -1.68488145e-01, -2.43736915e-02, 1.71931591e-02, 8.73959363e-02, -4.72040251e-02, -5.22271767e-02, 6.38323724e-02, -9.38716978e-02, -3.41966391e-01, -8.17023292e-02, -7.34338537e-02, -1.87136158e-02, -6.80101886e-02, -1.48000838e-02, 5.80909960e-02, -2.96405017e-01, -6.08926639e-02, 8.38592462e-03, 1.32774189e-01, 4.55581211e-03, -2.34010592e-02, 1.49530292e-01, 5.38847893e-02,-2.10960388e-01, 1.39128147e-02, 6.67495206e-02, 2.45740220e-01, 8.81331116e-02, 6.46145344e-02, 9.02941450e-04, -6.67425022e-02, 4.26766239e-02, -2.11339638e-01, 5.34121394e-02, 9.61864144e-02, 1.58464059e-01, 1.95462219e-02, 2.70930771e-02, -2.28668928e-01, 2.26574130e-02, 8.68059844e-02, -1.28633365e-01, 6.16969764e-02, 1.90359522e-02, -8.03845823e-02, -4.63765450e-02, -7.58518428e-02, 2.70034462e-01, 9.44757462e-02, -1.13836281e-01, -1.11676559e-01, 1.55629322e-01, -6.52169287e-02, -2.02764720e-02, 6.20288476e-02, -1.21912345e-01, -1.76230431e-01, -3.07087660e-01, 2.91486531e-02, 4.29091394e-01, 6.90438598e-02, -2.28306457e-01, 1.64378695e-02, -6.08395673e-02, -1.88289210e-03, 1.00612335e-01, 1.22119285e-01, 1.39086451e-02, 3.03243343e-02, -1.04687579e-01, 1.33531559e-02, 2.11238876e-01, -4.44081649e-02, -3.11685186e-02, 1.73644185e-01, -6.47547990e-02, 4.73273918e-02, -3.30260582e-02, -9.26901028e-03, -5.27807139e-02, -9.27131623e-06, -1.13007613e-01, -4.11117077e-02, -1.04314936e-02, -4.52113450e-02, -1.02403402e-01, 1.09692454e-01, -1.87846035e-01, -1.37371868e-02, 2.76893359e-02, -4.26322147e-02, -2.74904221e-02, 5.59798665e-02, -1.30586892e-01, -8.86572972e-02, 1.11441813e-01, -2.68283188e-01, 2.48056725e-01, 2.26278633e-01, -4.24992405e-02, 1.13888748e-01, 1.11040711e-01, 5.22921979e-02, 3.40353176e-02, -3.94822769e-02, -1.96606800e-01, -6.52438253e-02, 9.18982103e-02, -4.92494889e-02, 5.85471354e-02, -1.99707597e-02

Then:

$$d_{face} = \sqrt{\begin{array}{c} \left(-4.30667996-(-1.25418440)\right)^2{}_1 + (-2.28661392-8.47699940)^2{}_2 + \\ (4.41910885-1.29728168)^2{}_3 + \left(-6.76112324-(-8.97089690)\right)^2{}_4 + \\ \ldots + \left(-6.29274026-(4.92494889)\right)^2{}_{126} + (8.91835839-5.85471354)^2{}_{127} + \\ (1.40212625-(-1.99707597))^2{}_{128} \end{array}}$$

$$d_{face} \cong 0.544253377400266$$

Calculate Similarity Percentage:

$$Similarity\ Percentage = \left(1 - \frac{d_{face}}{d_{max}}\right) \times 100 + 25$$

$$Similarity\ Percentage = \left(1 - \frac{0.544253377400266}{1.0}\right) \times 100 + 25 = \mathbf{70.57466226}$$

```
No match found. Similarity: 70.40%
```
≅ **70.57466226**

Result: Similarity percentage of two different images is 70.57% which is beyond the accepted threshold and is True Negative

4.2 Enhancing Haar Cascade Algorithm gray scaling method with face_encoding and RGB images

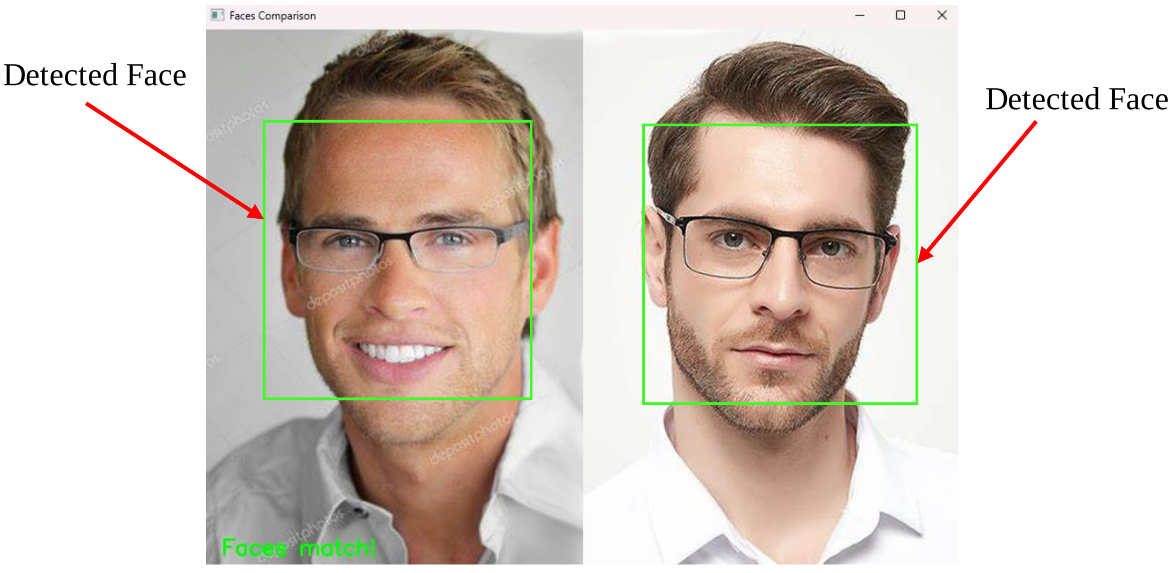
Applying the solutions from chapter 3, the researchers will conduct testing and comparison between the Haar Cascade Algorithm and Enhanced Haar Cascade Algorithm when it comes to similarity measurement (faces sharing similar attributes)

Addressing Statement of the Problem no.2

Haar Cascade gray scaling method

- Haar cascade covert image/s into grayscale for face detection which can result to a limited data extraction in the image since the histogram matching will rely on gray scaled image.
- In the example below, 2 different images sharing a same attribute will be processed by Haar Cascade Algorithm with gray scaling method and LBPH, and it will determine the similarity rate between the 2 images.

Example:



```
Similarity percentage: 89.29289786990967
```

Figure 18 Haar Cascade with images sharing similar attributes

Figure 18 shows a similarity percentage of 89.29% even though the images are significantly different from each other, but they both feature individuals wearing glasses.

This high similarity score results from the Haar Cascade's reliance on grayscale processing and LBPH that uses the same gray scaled image which can emphasize certain features like glasses while overlooking more distinguishing facial characteristics. Consequently, the algorithm misinterprets these shared attributes as an indication of a match, leading to a False Positive result. This underscores the problem of the Haar Cascade method when used with grayscale images and LBPH, as it may not capture every detail necessary for accurate face recognition, particularly in cases where non-facial features, such as glasses or other accessories, dominate the detection process.

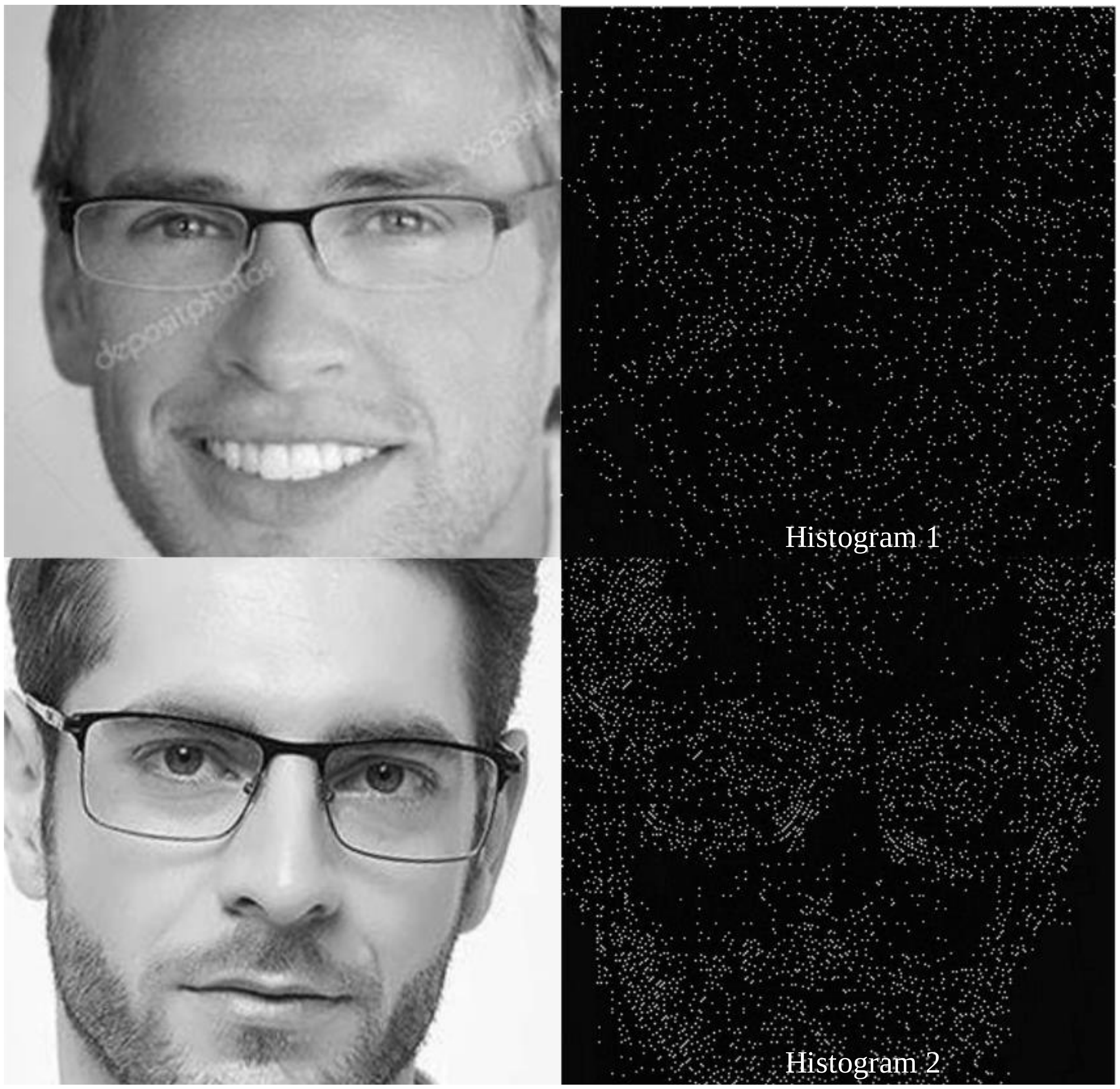


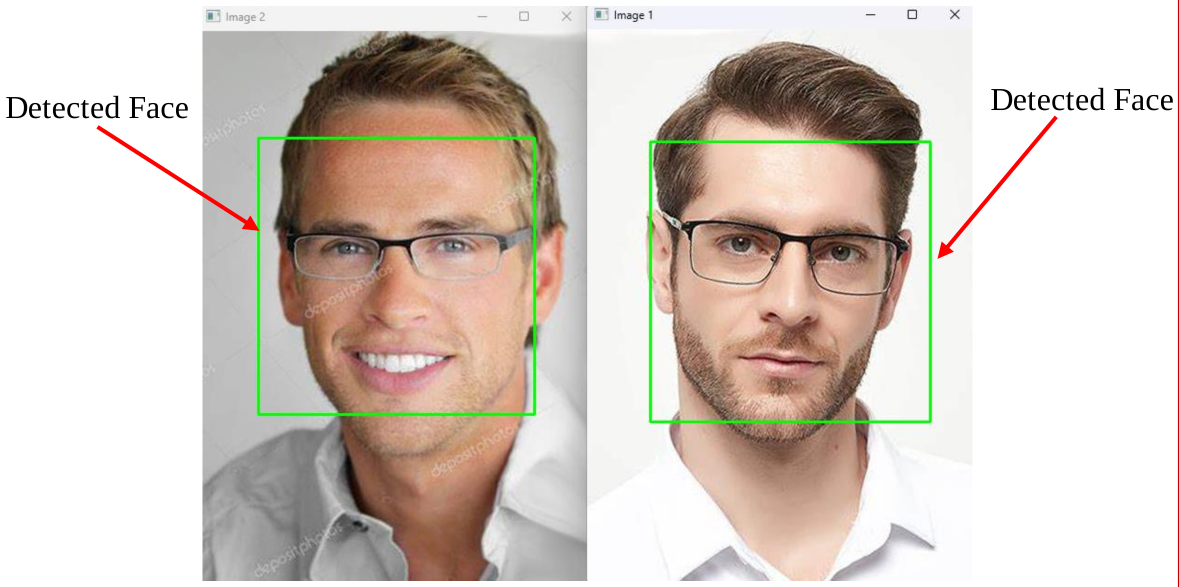
Figure 19 Haar Cascade grayscale and LBP Histogram

Figure 19 shows the result of Haar Cascade gray scaling method together with the conversion to LBPH for face recognition. As the Haar Cascade relies on the black and white images for face detection the LBPH will also use the gray scaled images to create a Histogram for each image.

Enhanced Haar Cascade with face_encoding and RGB images

- The enhanced approach uses the Haar Cascade algorithm in combination with face_encoding from the face_recognition library, which processes images in their full RGB color format rather than converting them to grayscale. This allows for a richer data extraction, as color information is preserved, enhancing the accuracy of face recognition by considering a broader range of facial features.
- In the example below, two different images sharing a similar attribute will be processed by the enhanced Haar Cascade Algorithm with the face_encoding method. The RGB images will be used to determine the similarity rate between the two images.

Example:



```
No match found. Similarity: 46.36%
```

Figure 20 Enhanced Haar Cascade with images sharing similar attributes

Figure 20 shows a similarity percentage of 46.36% when comparing the same images from Figure 4.3 after applying the RGB method, though the approach is different as it uses the Enhanced Haar Cascade with face_encoding and RGB images. The result correctly returns a True Negative since the images are of two different individuals who only share the attribute of wearing glasses. This lower similarity score highlights the improved accuracy of the enhanced approach, as it considers the full range of facial features by processing the images in RGB. Unlike the grayscale method, which can be misled by shared attributes like glasses, the use of face_encoding ensures that the algorithm better distinguishes between unique facial characteristics, reducing the likelihood of False Positives.

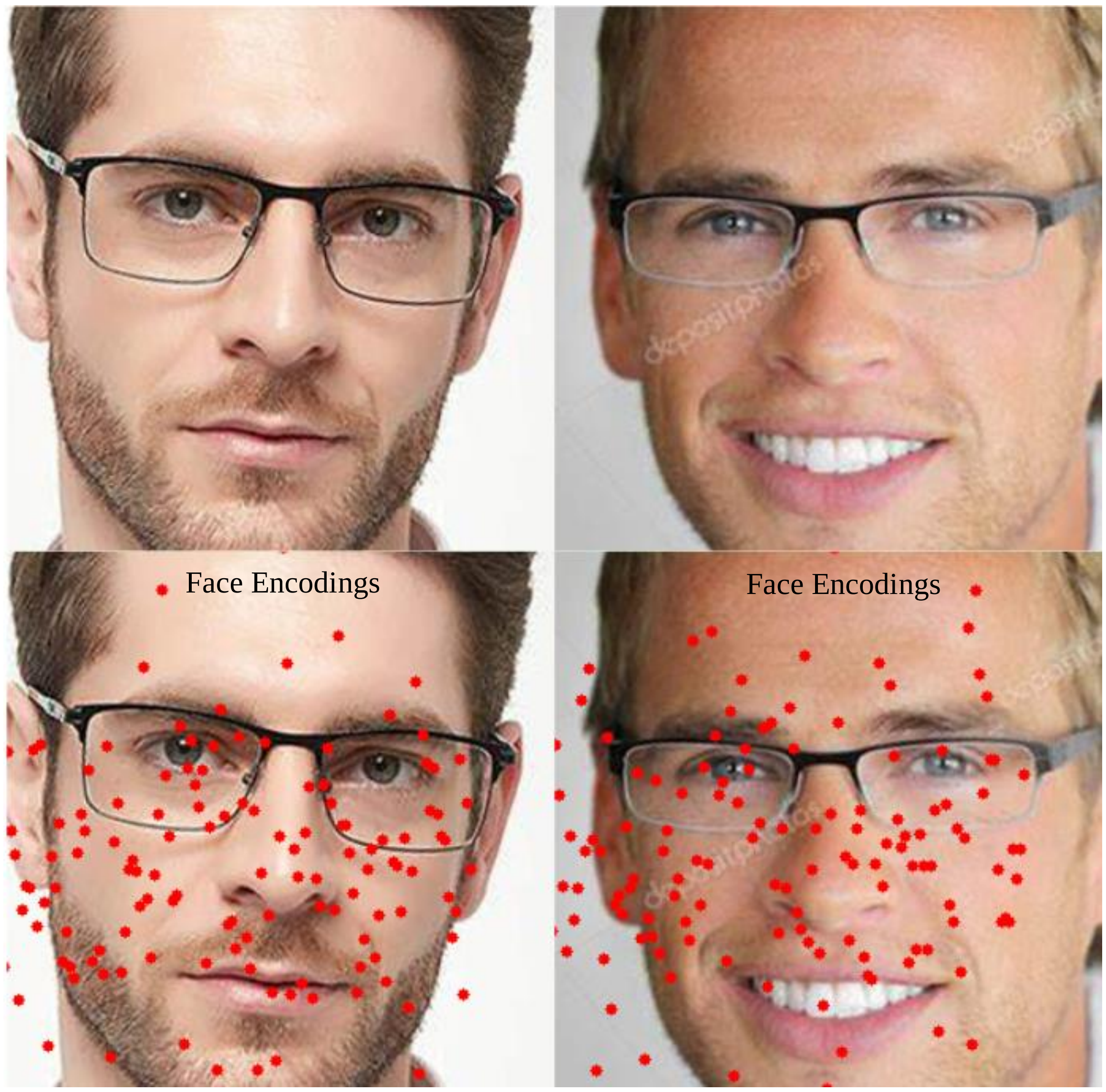


Figure 21 Enhanced Haar Cascade with face_encoding and RGB

Figure 21 shows the face encoding with RGB. The face encoding relies on RGB images to select the 128-dimensional encoding instead of using gray scaled image for texture base encoding.

4.3 Haar Cascade Algorithm by tweaking parameters scaleFactor, minNeighbors, and minSize using a logic process and filtering

Applying the solutions from chapter 3, the researchers will conduct testing and comparison between the Haar Cascade Algorithm and Enhanced Haar Cascade Algorithm when it comes to face detection by tweaking parameters scaleFactor, minNeighbors, and minSize using a logic process

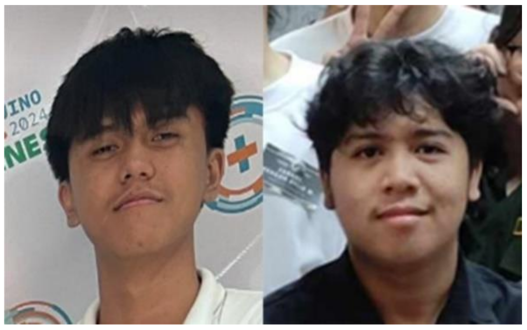
Addressing Statement of the Problem no.3

Haar Cascade Algorithm parameter setting:

- scaleFactor = 1.1
- minNeighbors = 5
- minSize = 30, 30

In the context of face detection using Haar Cascade, a **scaleFactor** of 1.1 means that the image size is reduced by 10% at each step of the detection process, allowing the algorithm to detect faces of various sizes in the image. The **minNeighbors** parameter is set to 5, meaning that a face must have at least 5 neighboring rectangles (positive detections) grouped together to be considered valid, which helps in reducing false positives. Finally, a **minSize** of (30, 30) specifies that the smallest face the detector will attempt to find must be at least 30x30 pixels, allowing the algorithm to ignore regions of the image smaller than this size.

Example:



```
Error: No face detected in the image.
```

Figure 22 Haar Cascade face detection issue

Figure 22 highlights a facial detection error when comparing Image 1 and Image 2. Given that Image 2 has been successfully processed in previous tests, the issue likely originates from Image 1. This error may be attributed to the default parameter settings used in the detection algorithm, specifically with scaleFactor = 1.1, minNeighbors = 5, and minSize = (30, 30).

Example:

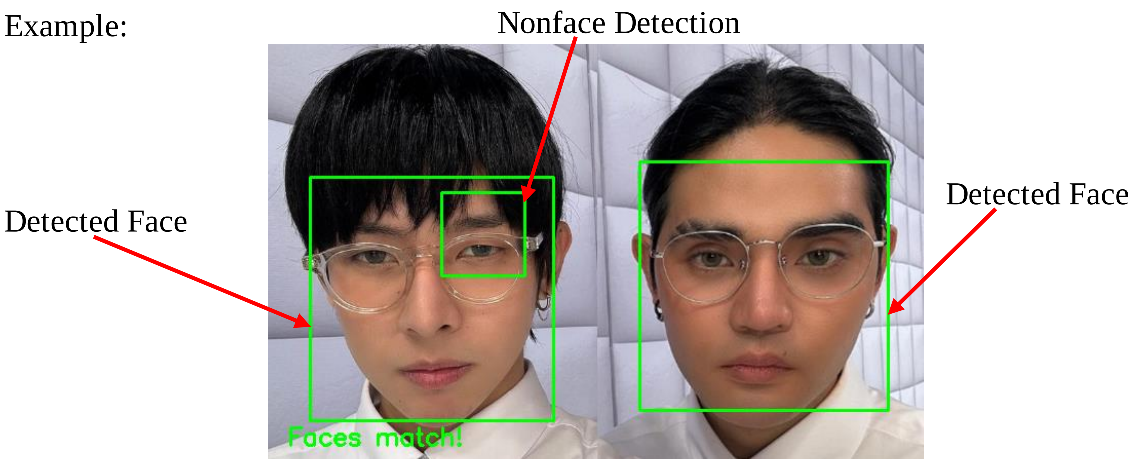


Figure 23 Haar Cascade non-face detection

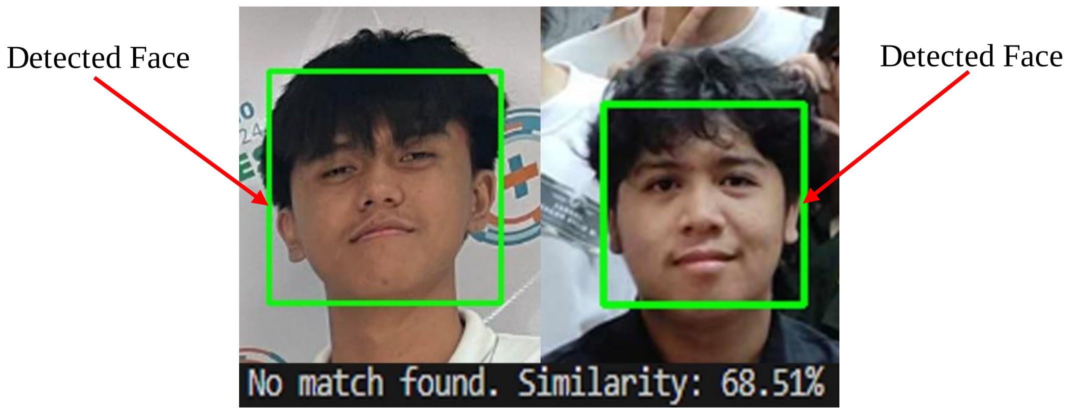
Figure 23 shows a false positive detection of a non-faced object in the given image. The green boxes represent the face that are detected in the image, there are only 2 faces but the result returns 3 boxes wherein one of the boxes is a false positive as it considers the eye as a face.

Using a scaleFactor of 1.1 means that the image is downscaled by 10% at each step, which is typically sufficient for detecting faces of various sizes. However, if the face in Image 1 is either too small or not prominent enough in the initial scaling steps, the detector may fail to recognize it as a face. The minNeighbors parameter set to 5 requires at least five rectangles to be detected around a region for it to be confirmed as a face. If the face in Image 1 doesn't generate enough strong detections, it might be missed. Finally, a minSize of (30, 30) ensures that any face smaller than 30x30 pixels is ignored by the algorithm. If the face in Image 1 is smaller than this threshold or too ambiguous in shape, the algorithm may not detect it, leading to the observed error. These settings, while generally effective, may not be ideal for all images, particularly if the face size or quality in the image doesn't meet the expected parameters, resulting in missed detections.

Enhanced Haar Cascade Algorithm parameter setting:

For the enhanced Haar Cascade Algorithm parameter, instead of using the traditional usage of changing the baseline values we will create a logic process where it will iterate thru several parameter values and filter the result

Example:

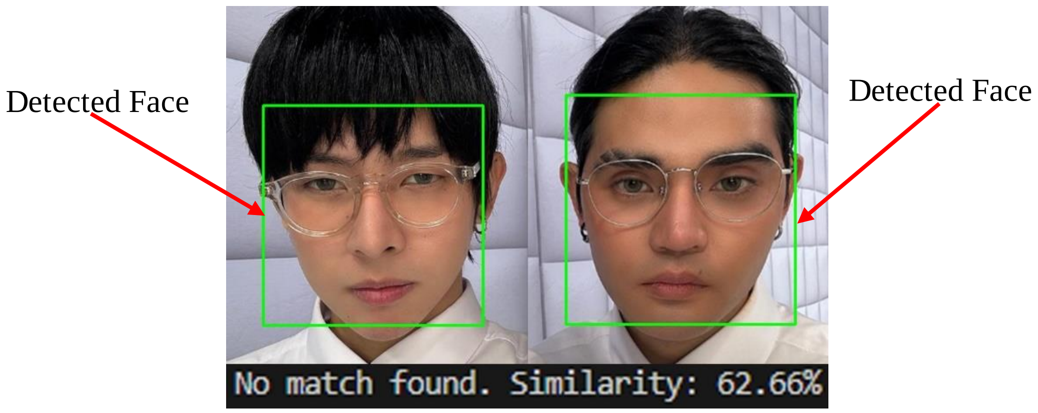


```
No match found. Similarity: 68.51%
```

Figure 24 Enhanced face detection process

Figure 24 shows that after enhancing process of face detection where logical process and face filtering are involved, it can now detect faces in complex image which is not detectible before when relying on the baseline values of the parameters.

Example:



```
No match found. Similarity: 62.66%
```

Figure 25 Enhanced face detection with filtering

Figure 25 shows that the previous non-face detection is gone due to the filtering process included in the face detection process. Instead of just detecting the face, it also added a condition if that detected face center is closer to the center of the image.

4.4 Comparison of Haar Cascade Algorithm Vs. Enhanced Haar Cascade Algorithm

Table 1 Comparison between Haar Cascade and Enhanced Haar Cascade (Same Person)

Table 1 show the comparison between the same person using the original and enhanced Haar Cascade Algorithm

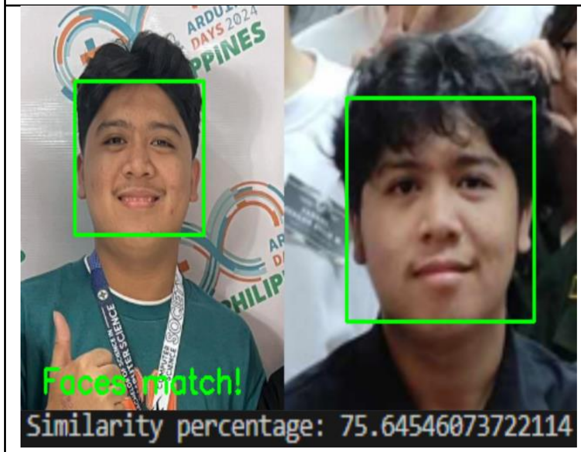
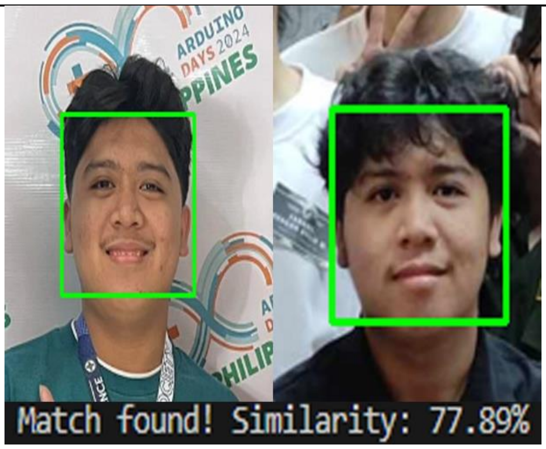
Haar Cascade Algorithm	**Enhanced Haar Cascade Algorithm**
 Similarity percentage: 75.64546073722114	 Match found! Similarity: 77.89%
The Haar Cascade shows a similarity percentage of 75.65% which shows a good example of facial recognition without any added modification.	The Enhanced Haar Cascade shows a similarity percentage of 77.89% which is a little higher than the previous comparison
There is a 2.24% increase in similarity rate upon implementing the face_recognition to Haar Cascade.	

Table 2 Comparison between Haar Cascade Algorithm and Enhanced Haar Cascade Algorithm (Different Person)

Table 2 show the comparison between the different person using the original and enhanced Haar Cascade Algorithm

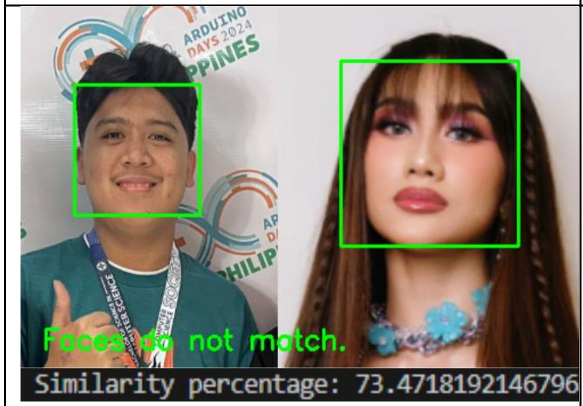
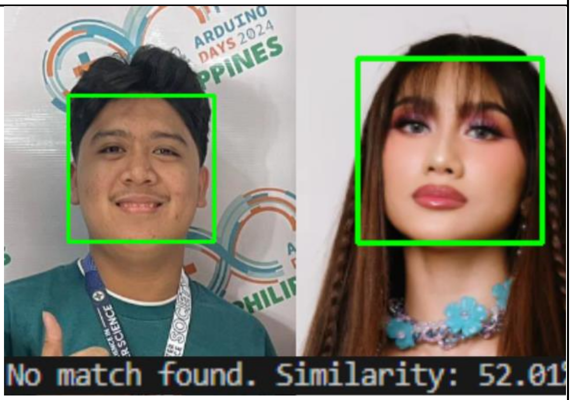
Haar Cascade Algorithm	**Enhanced Haar Cascade Algorithm**
 Faces do not match. Similarity percentage: 73.4718192146796	
The Haar Cascade shows a similarity percentage of 73.47% for comparing different person which state that it is a True Negative.	The Enhanced Haar Cascade shows a similarity percentage of 52.01% when comparing different person using a enhanced approach which also return as True Negative.
The enhanced approach is 21.36% lower than when comparing 2 different images and calculating their similarity percentage, it shows that the enhanced haar cascade algorithm is more accurate at calculating images with 2 different people.	

Table 3 Comparison between Haar Cascade Algorithm and Enhanced Haar Cascade Algorithm (Different Person - 2nd Trial)

Table 3 shows the comparison between the different person using the original and enhanced Haar Cascade Algorithm, in this case both images share some similarities particularly in lighting.

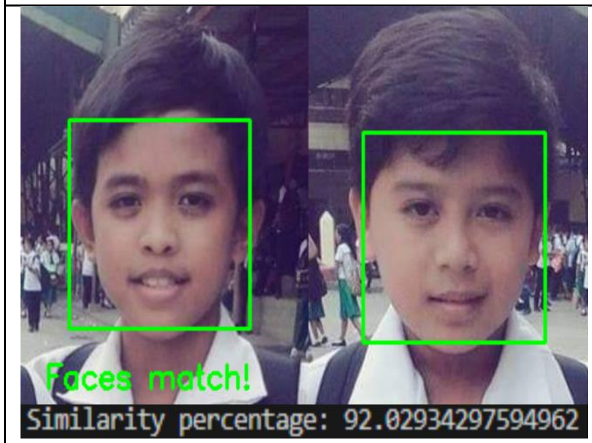
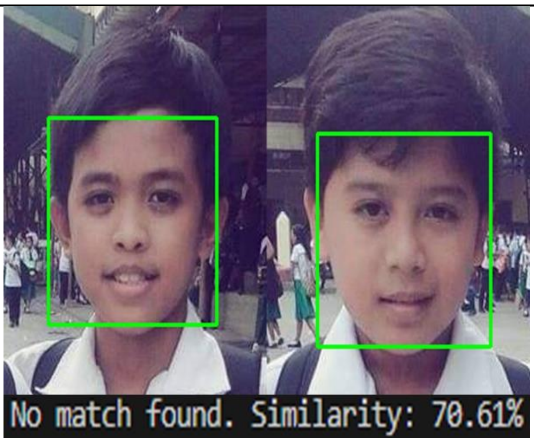
Haar Cascade Algorithm	Enhanced Haar Cascade Algorithm
	
In the 2nd trial on Haar Cascade using different person, it shows a similarity of 92.03% which is False Positive	While in the Enhanced Haar Cascade, the comparison between different people resulted in a 70.61% similarity rate (True Negative) and not a match considering the 75% acceptance threshold.
The Haar Cascade reaches a 92.03% similarity in comparing different person with slightly similar attributes such as lighting and facial features. While the Enhanced Haar Cascade resulted in a 70.61% (still close to the threshold but not accepted), and still a True Negative	

Table 4 Comparison between Haar Cascade Algorithm and Enhanced Haar Cascade Algorithm (Same Person - Inverted)

Table 4 show the comparison between the same person using the original and enhanced Haar Cascade Algorithm, but the images are just inverted.

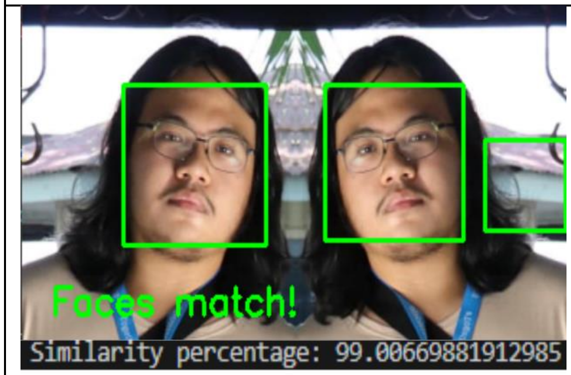
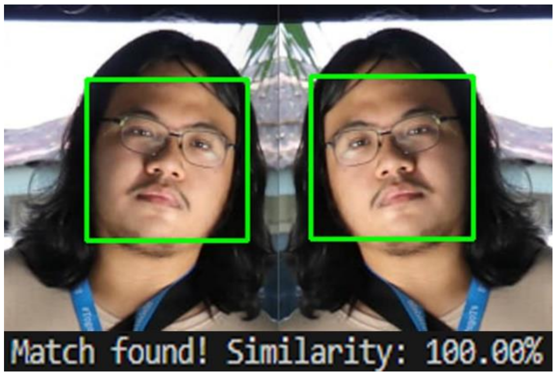
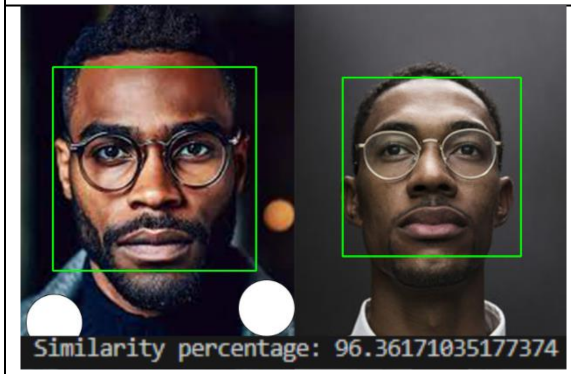
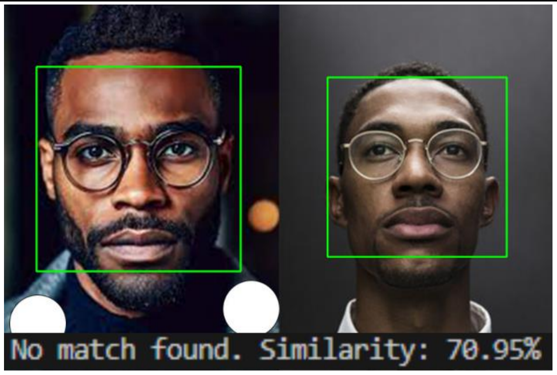
Haar Cascade Algorithm	**Enhanced Haar Cascade Algorithm**
	
When comparing similar image but inverted, the Haar Cascade shows a 99.00% similarity which is almost 100	At the same time, in the Enhanced Haar Cascade Algorithm, a similarity percentage of 100% was calculated in the inverted image.
While the Haar Cascade Algorithm detected a 99.00% similarity between the and inverted images, the Enhanced Haar Cascade Algorithm identified a 100% similarity in the same scenario. However, the Haar Cascade Algorithm also mistakenly detected a non-face area in the image which also indicate false positive face detection.	

Table 5 Comparison between Haar Cascade Algorithm and Enhanced Haar Cascade Algorithm (Person sharing similar attributes)

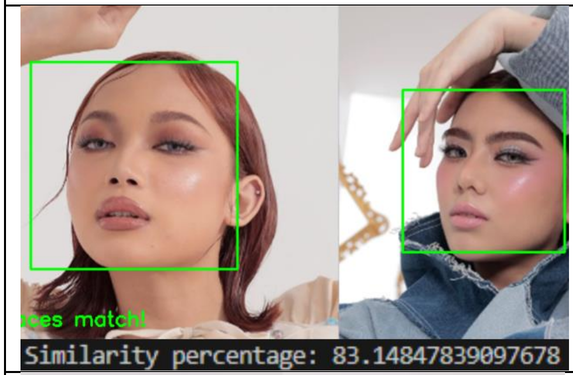
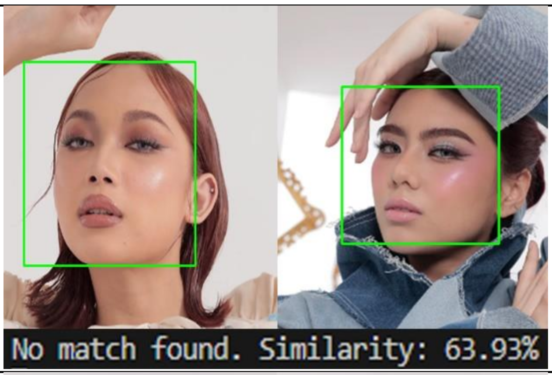
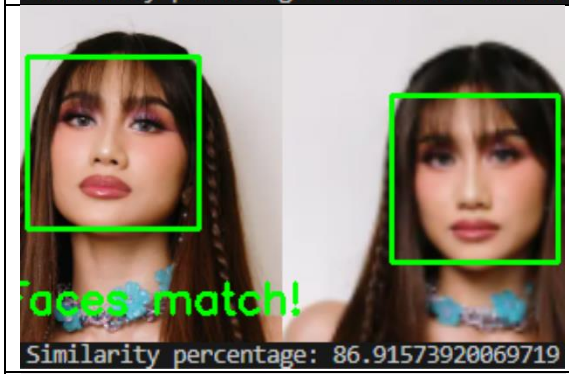
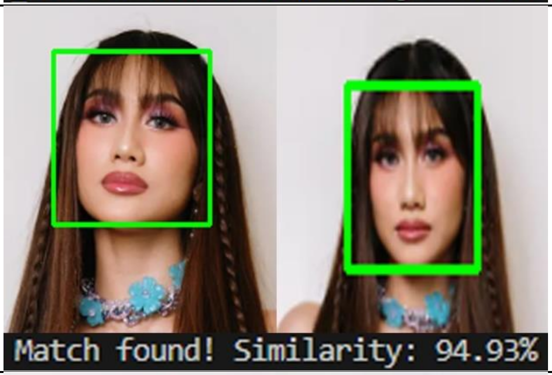
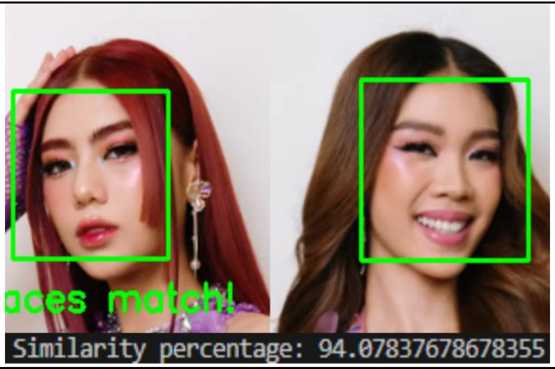
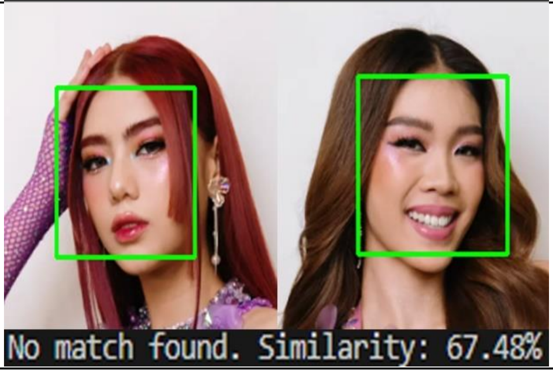
Table 5 show the comparison between the different person using the original and enhanced Haar Cascade Algorithm, both images also share similarity in skin tone and glasses.

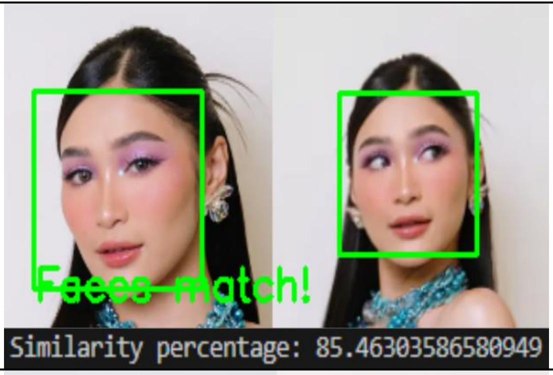
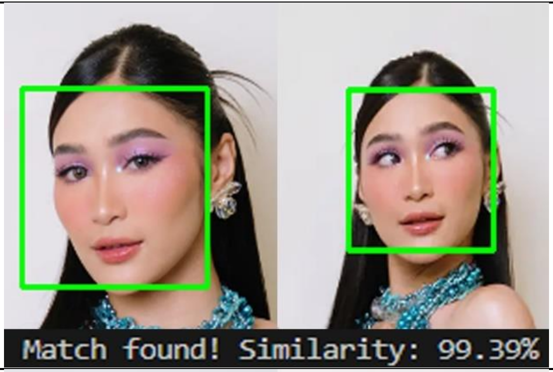
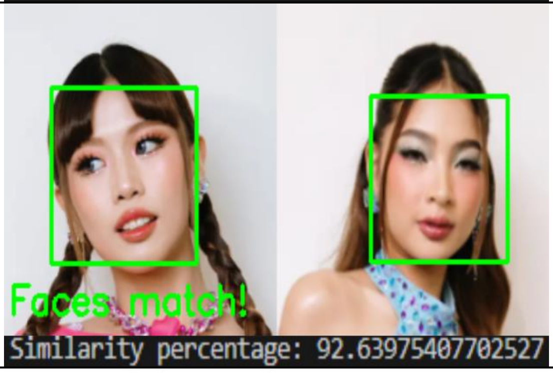
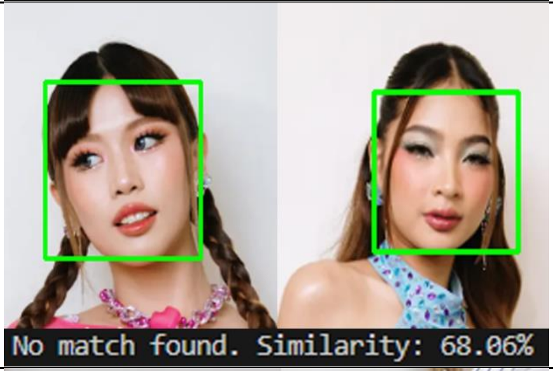
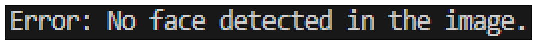
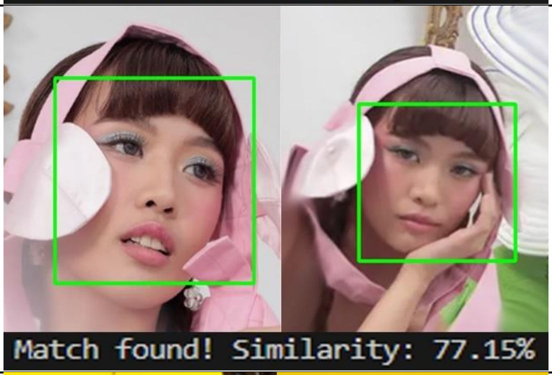
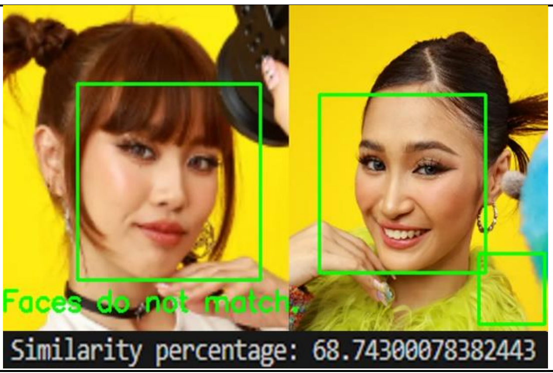
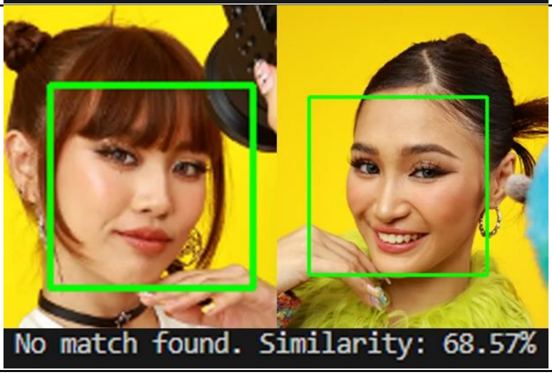
Haar Cascade Algorithm	**Enhanced Haar Cascade Algorithm**
 Similarity percentage: 96.36171035177374	 No match found. Similarity: 70.95%
In comparing images sharing similar attributes (wearing glasses), the Haar Cascade calculated a 96.36% of similarity between 2 different people which results to False Positive.	While the Enhanced Haar Cascade calculated a 70.95% of similarity with the same image, even that is it close to the threshold, it still results to a True Negative outcome.
When comparing images of different individuals wearing glasses, the Haar Cascade Algorithm produced a high similarity score of 96.36%, leading to a False Positive result. Conversely, the Enhanced Haar Cascade Algorithm calculated a lower similarity of 70.95%, which, although close to the threshold, correctly identified the images as not matching, resulting in a True Negative. This demonstrates that the modifications improve the algorithm's ability to distinguish between similar yet distinct faces, reducing the likelihood of False Positives.	

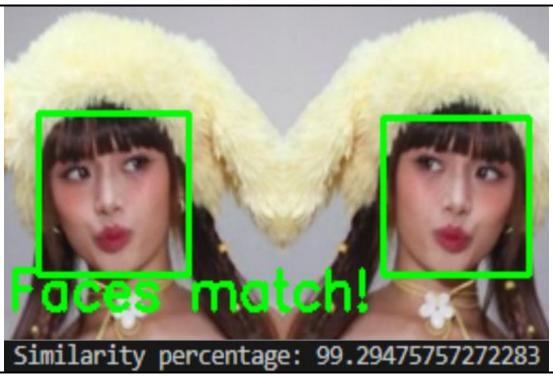
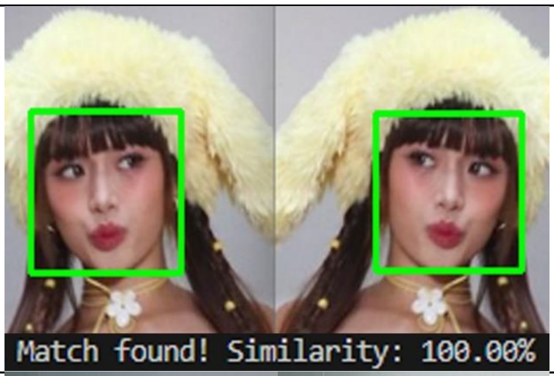
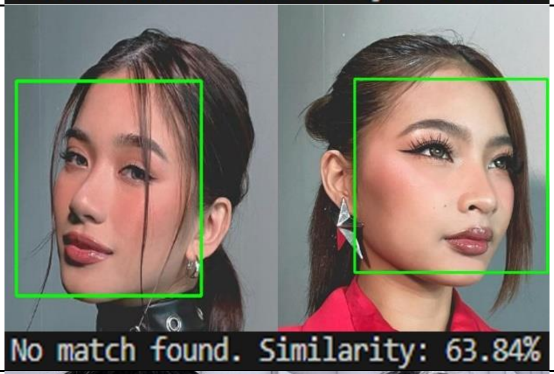
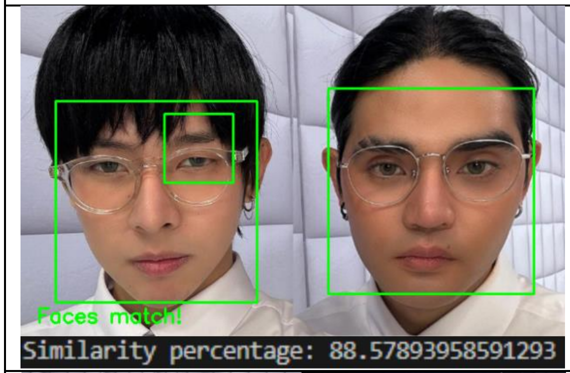
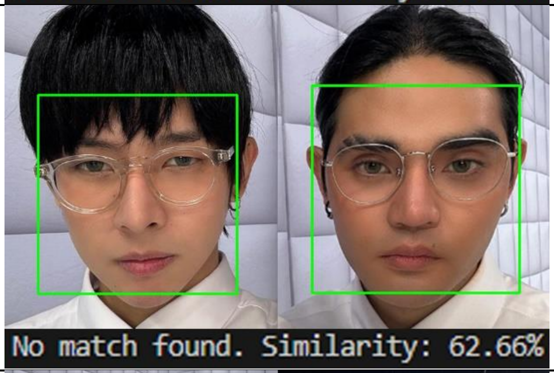
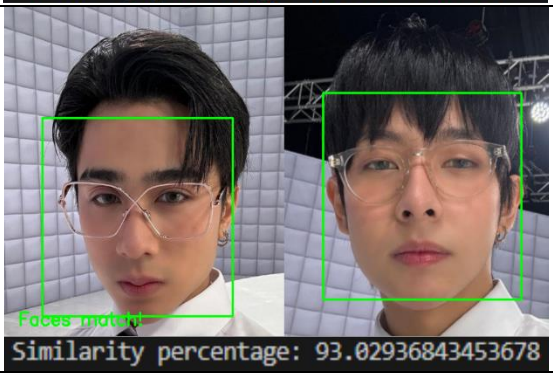
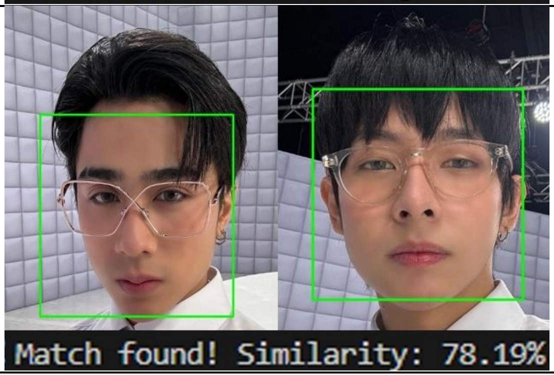
4.5 Other comparison between the Haar Cascade Algorithm Vs. Enhanced Haar Cascade Algorithm

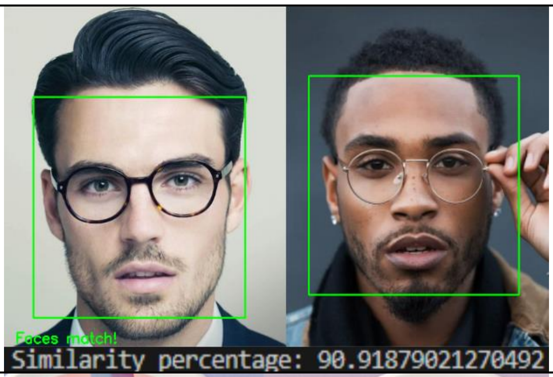
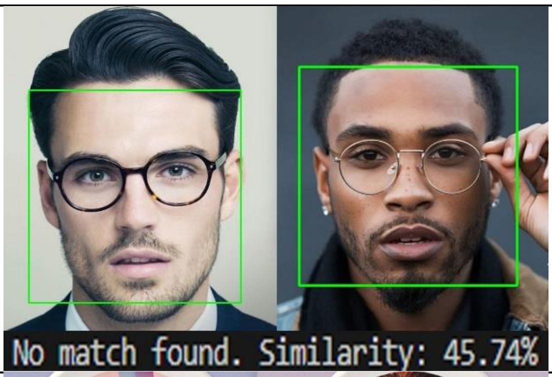
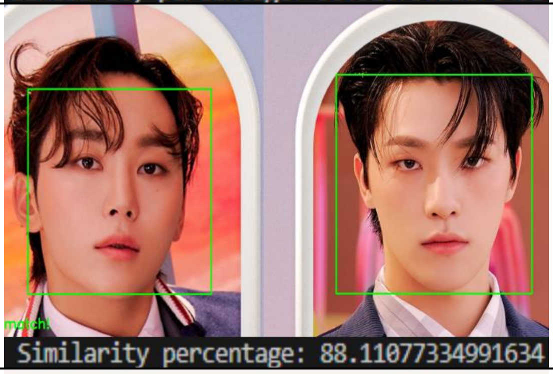
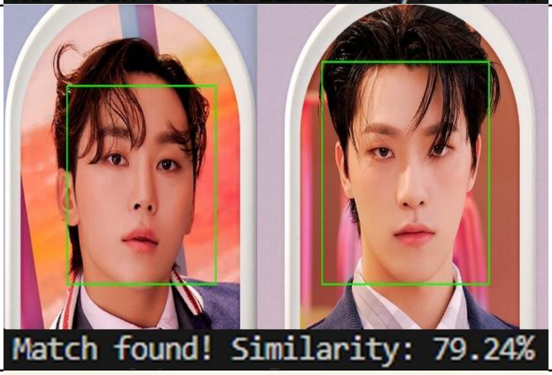
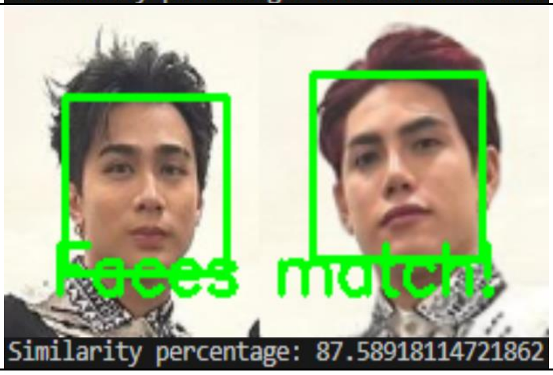
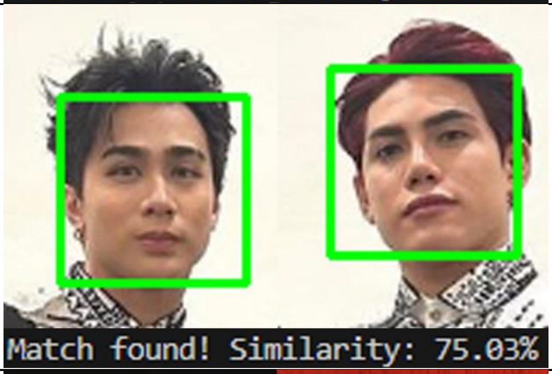
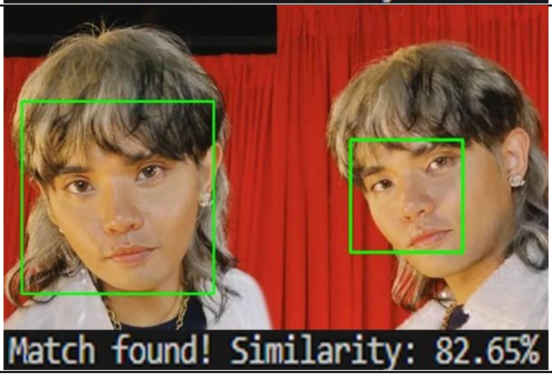
Table 6 15 Comparison between Haar Cascade and Enhanced Haar Cascade Algorithm

In this section, we will present various outputs comparing the Haar Cascade Algorithm with the Enhanced Haar Cascade Algorithm. However, we will not provide a detailed discussion of each output, as this will allow for a more straightforward observation and comparison of the differences between the two methods.

Haar Cascade Algorithm	Enhanced Haar Cascade Algorithm
 ces match! Similarity percentage: 83.14847839097678	 No match found. Similarity: 63.93%
 aces match! Similarity percentage: 86.91573920069719	 Match found! Similarity: 94.93%
 aces match! Similarity percentage: 94.07837678678355	 No match found. Similarity: 67.48%

Haar Cascade Algorithm	Enhanced Haar Cascade Algorithm
 Similarity percentage: 85.46303586580949	 Match found! Similarity: 99.39%
 Similarity percentage: 92.63975407702527	 No match found. Similarity: 68.06%
 Error: No face detected in the image.	 Match found! Similarity: 77.15%
 Similarity percentage: 68.74300078382443	 No match found. Similarity: 68.57%

Haar Cascade Algorithm	Enhanced Haar Cascade Algorithm
 Similarity percentage: 99.29475757272283	 Match found! Similarity: 100.00%
Error: No face detected in the image.	 No match found. Similarity: 63.84%
 Similarity percentage: 88.57893958591293	 No match found. Similarity: 62.66%
 Similarity percentage: 93.02936843453678	 Match found! Similarity: 78.19%

Haar Cascade Algorithm	Enhanced Haar Cascade Algorithm
 Similarity percentage: 90.91879021270492	 No match found. Similarity: 45.74%
 Similarity percentage: 88.11077334991634	 Match found! Similarity: 79.24%
 Similarity percentage: 87.58918114721862	 Match found! Similarity: 75.03%
Error: No face detected in the image.	 Match found! Similarity: 82.65%

4.6 Performance evaluation using Confusion Matrix Analysis (Test 1)

Table 7 Confusion Matrix Analysis (face matching) of Table 6

Table 7 shows a tally of scores from Haar Cascade and Enhanced Haar Cascade algorithm in terms of face matching using the conditions:

- (TP): Both faces are perfectly matched.
- (FP): Different face but resulted to match or there's a face but not detected
- (TN): Different face and returned as not match.
- (FN): Same face but resulted as not match.

Face matching using:	TP	FP	TN	FN
Haar Cascade Algorithm	3	12	0	0
Enhanced Haar Cascade Algorithm	5	3	7	0

Table 8 Confusion Matrix Analysis result (face matching)

Table 8 shows the result of the confusion matrix analysis of the 15 comparisons in terms of face matching, it clearly shows that the Enhanced approach has a higher score in Accuracy, Precision, and F1 Score with both being equal in Recall.

Face matching using:	Accuracy	Precision	Recall	F1 Score
Haar Cascade Algorithm	20%	20%	100%	33.33%
Enhanced Haar Cascade Algorithm	80%	62.5%	100%	76.92%

Table 9 Confusion Matrix Analysis (face detection) of Table 6

Table 9 shows a tally of scores from Haar Cascade and Enhanced Haar Cascade algorithm in terms of face detection using the conditions:

- (TP): There's 1 face in the image.
- (FP): There are more than 1 faces in the image.
- (TN): There's no face in the image.
- (FN): There's 1 face in the image but it wasn't detected.

Face detection using:	TP	FP	TN	FN
Haar Cascade Algorithm	10	2	0	3
Enhanced Haar Cascade Algorithm	15	0	0	0

Table 10 Confusion Matrix Analysis result (face detection)

Table 10 shows the result of the confusion matrix analysis of the 15 comparisons in terms of face detection after applying the Enhanced process of face detection with filtering. It surprisingly achieved a 100% score in for Accuracy, Precision, Recall, and F1 Score which signify the improvement made from the Haar Cascade Algorithm.

Face detection using:	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Haar Cascade Algorithm	66.66%	83.33%	76.92%	80%
Enhanced Haar Cascade Algorithm	100%	100%	100%	100%

4.7 Performance evaluation using Confusion Matrix Analysis (Test 2)

In this part, a total of 301,950 comparisons will be made using the 550 images collected, 526 of the images contains 1 face only and complex background, the remaining 24 images are non-face image.

Table 11 Confusion Matrix Analysis result (Test 2)

Table 11 shows the result of the 301,950 comparisons, it clearly indicates that the enhanced Haar Cascade algorithm significantly outperforms the standard version across all metrics except recall, where it maintains comparable performance. The stark improvement in accuracy and precision suggests that the modifications have effectively reduced the number of false positives, leading to a more trustworthy face detection system. The high recall values in both algorithms imply that the detection coverage is comprehensive, ensuring that most faces are captured. The enhancement in the F1 Score for the enhanced algorithm underscores the balanced improvement across both precision and recall, making it a more robust and reliable option for practical applications.

	Haar Cascade Algorithm			
	TP	**FP**	**TN**	**FN**
Face Matching	504	265234	12504	14
Face Detection	250500	33810	17424	216
	Accuracy	**Precision**	**Recall**	**F1 Score**
Face Matching	4.674832	0.18966	97.2973	0.378583
Face Detection	88.73125	88.10805	99.91385	93.64031
	Accuracy	**Precision**	**Recall**	**F1 Score**
TOTAL SCORE	**46.70%**	**44.15%**	**98.61%**	**47.01%**
	Enhanced Haar Cascade Algorithm			
	TP	**FP**	**TN**	**FN**
Face Matching	512	1186	282906	18
Face Detection	276150	8472	17328	0
	Accuracy	**Precision**	**Recall**	**F1 Score**
Face Matching	99.57698	30.15312	96.60377	45.9605
Face Detection	97.19424	97.02342	100	98.48923
	Accuracy	**Precision**	**Recall**	**F1 Score**
TOTAL SCORE	**98.39%**	**63.59%**	**98.30%**	**72.23%**
	Accuracy	**Precision**	**Recall**	**F1 Score**
Haar Cascade Algorithm	46.70%	44.15%	98.61%	47.01%
Enhanced Haar Cascade Algorithm	98.39%	63.59%	98.30%	72.23%

Chapter Five

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

- The researchers discovered that incorporating the face recognition library into the Haar Cascade algorithm significantly improved image comparison accuracy, even under challenging conditions. The enhancement achieved a 98.39% in accuracy, a 21.39% from the 46.70% to 77.00% baseline accuracy rate.
- Additionally, by adding face encoding with RGB images, the accuracy of facial matching increased substantially, as more detailed data was extracted, even when the images shared similar attributes. The enhanced algorithm has a 99.58% accuracy when comparing images with similar attributes. A 22.58% increase from the baseline accuracy rate of 46.70% to 77.00%.
- Furthermore, face detection accuracy improved after modifying the baseline parameters with logical and filtering processes, reducing the detection of non-face objects in the images. It achieved a 97.19% accuracy rate, an increase of 8.46% from the previous face detection accuracy rate of 88.73%.

5.2 Recommendations

- The researchers recommend using the Enhanced Haar Cascade Algorithm in face recognition systems to improve accuracy, especially in challenging conditions.
- As the Enhanced Haar Cascade Algorithm uses RGB conversion, it can used even in black and white images making more reliable in face recognition.
- It is recommended to regularly refine detection parameters and incorporate logical filtering processes to enhance detection accuracy and reduce false positives from non-face objects.

LIST OF REFERENCES

Pandey, A., Choudhary, D., Agarwal, R., Shrivastava, T., & K. (2022, January 1). Face detection using Haar cascade classifier. Social Science Research Network. https://doi.org/10.2139/ssrn.4157631

Preeti Saini, Bharti Nagpal, Sheersh Kaushik, & Purnima Gupta. (2019). Face Recognition in Real Time Video using Haar Cascade Classifier. . http://bvicam.in/INDIACom/news/INDIACom%202019%20Proceedings/Main/papers/106.pdf

Legaspi, R. C. (2023). Localizing Face Recognition with Haar-Cascade Classifier and LBPH using Python. https://www.semanticscholar.org/paper/Localizing-Face-Recognition-with-Haar-Cascade-and-Legaspi/74df2b542646c514be9febbef3561f042af0a797

Gede Susrama Mas Diyasa, & *, Alfian Hendika Putra. (2022). Feature Extraction for Face Recognition Using Haar Cascade Classifier. . https://dpnet.org.np/public/uploads/files/673-Article%20Text-2075-1-10-20220517%202023-12-21%2010-10-18.pdf

Wattamwar, S., & Mate, R. (2021, October 29). Optimal Face Recognition System using Haar Classifier. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9645879

Safiullina, L. Kh., & Gabdullin, A. Sh. (2021, November 9). Face recognition in biometric systems using Haar cascade classification. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9653460

Cheol-Ho Choi, Junghwan Kim, Jongkil Hyun, & Younghyeon Kim. (2022, March). Face Detection Using Haar Cascade Classifiers Based on Vertical Component Calibration. . https://doi.org/10.22967/HCIS.2022.12.011

Yustiawati, R., Husni, N. L., Evelina, E., Rasyad, S., Lutfi, I., Silvia, A., Alfarizal, N., & Rialita, A. (2018, October). Analyzing Of Different Features Using Haar Cascade Classifier. 2018 International Conference on Electrical Engineering and Computer Science (ICECOS). https://doi.org/10.1109/icecos.2018.8605266

Siti Hashim, & Paul Mccullagh. (2023, March). Face detection by using Haar Cascade Classifier. . https://doi.org/10.31185/wjcm.109

Hapsari, D. Y. P. (2018, September 1). Face detection using haar cascade in difference illumination. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/8549752

Gangopadhyay, I., Chatterjee, A., & Das, I. (2019). Face Detection and Expression Recognition Using Haar Cascade Classifier and Fisherface Algorithm. https://www.semanticscholar.org/paper/Face-Detection-and-Expression-Recognition-Using-and-Gangopadhyay-Chatterjee/48e5638af232722ff0076f531ae7e7134e556400

K .H. Wanjale, Amit Bhoomkar, Ajay Kulkarni, & Somnath Gosavi. (2013, April). Use Of Haar Cascade Classifier For Face Tracking System In Real Time Video. . https://www.ijert.org/research/use-of-haar-cascade-classifier-for-face-tracking-system-in-real-time-video-IJERTV2IS4381.pdf

Zankruti Arya, & Vibha Tiwari. (2020, June). Automatic Face Recognition and Detection Using OpenCV, Haar Cascade and Recognizer for Frontal Face. Automatic Face Recognition and Detection Using OpenCV, Haar Cascade and Recognizer for Frontal Face. https://www.ijera.com/papers/vol10no6/Series-5/D1006051319.pdf

Ahmad, A. H., Saon, S., Mahamad, A. K., Darujati, C., Mudjanarko, S. W., Nugroho, S. B., & Hariadi, M. (2021, March 10). Real time face recognition of video surveillance system using haar cascade classifier. Indonesian Journal of Electrical Engineering and Computer Science. https://doi.org/10.11591/ijeecs.v21.i3.pp1389-1399

Mantoro, T., & Ayu, A. (2018, May 1). Multi-Faces recognition process using haar cascades and eigenface methods. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/8525935

Chinimilli, B. T., Anjali, T., Kotturi, A., Kaipu, V. R., & Mandapati, J. V. (2020, June 1). Face Recognition based Attendance System using Haar Cascade and Local Binary Pattern Histogram Algorithm. 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184). https://doi.org/10.1109/icoei48184.2020.9143046

Sharma, R., Ashwin, T. S., & Guddeti, R. M. R. (2018, November 4). A Novel Real-Time Face Detection System Using Modified Affine Transformation and Haar Cascades. Advances in Intelligent Systems and Computing. https://doi.org/10.1007/978-981-10-8639-7_20

Bairagi, R., Ahmed, R., & Tisha, S. A. (2021, September 2). A Real-time Face Recognition Smart Attendance System with Haar Cascade Classifiers. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9544872

Javed Mehedi Shamrat, F. M., Majumder, A., Antu, P. R., Barmon, S. K., Nowrin, I., & Ranjan, R. (2022). Human Face Recognition Applying Haar Cascade Classifier.

Pervasive Computing and Social Networking, 143–157. https://doi.org/10.1007/978-981-16-5640-8_12

Rapid object detection using a boosted cascade of simple features. (2001). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/990517

Cuimei, L., Zhiliang, Q., Nan, J., & Jianhua, W. (2017). Human face detection algorithm via Haar cascade classifier combined with three additional classifiers. Human Face Detection Algorithm via Haar Cascade Classifier Combined With Three Additional Classifiers. https://doi.org/10.1109/icemi.2017.8265863

Oluwaseun O. Adekola. (2023, March). A REVIEW OF EXISTING FACE DETECTION & RECOGNITION ALGORITHMS AND THE PERFORMANCE EVALUATION OF HAAR CASCADE ALGORITHM ON IMAGES USING OpenCV. A REVIEW OF EXISTING FACE DETECTION & RECOGNITION ALGORITHMS AND THE PERFORMANCE EVALUATION OF HAAR CASCADE ALGORITHM ON IMAGES USING OpenCV. https://doi.org/10.5281/zenodo.7768760

Singh, A., Herunde, H., & Furtado, F. (2020, June 1). Modified Haar-cascade model for face detection issues. DOAJ (DOAJ: Directory of Open Access Journals). https://doi.org/10.22105/riej.2020.226857.1129

Phuc, L. T. H., Jeon, H., Truong, N. T. N., & Hak, J. J. (2019, September 23). Applying the Haar-cascade Algorithm for Detecting Safety Equipment in Safety Management Systems for Multiple Working Environments. Electronics. https://doi.org/10.3390/electronics8101079

M.S. Minu, Kshitij Arun, Anmol Tiwari, & Priyansh Rampuria. (2020). Face_Recognition_System_Based_on_Haar_Cascade_Classifier. https://www.researchgate.net.

Shetty, A. B., B., D., Rebeiro, J., & R. (2021, November 1). Facial recognition using Haar cascade and LBP classifiers. Global Transitions Proceedings. https://doi.org/10.1016/j.gltp.2021.08.044

Orquez, P. X. M., & Sarin, J. O. (2013). Enhancement of Windows User Access Security using Multi-feature Face Detection and Recognition. https://ejournals.ph/article.php?id=6143

Ramos, A. L. A., Reyes, B. L., Nuevo, J. J., Avila, P. A., Bas, E. A., & Gonzales, P. J. M. (2019). FACIAL RECOGNITION PERFORMANCE BASED ON THE LIGHTING SET-UP MODELS APPLIED TO HOME SECURITY DOOR ACCESS USING PRINCIPAL COMPONENT ANALYSIS AND RASPBERRY PI CONTROLLER. https://ejournals.ph/article.php?id=14063

Ramos, A. L. A., Buenafe, P. A., Cabrales, E. K. C., Teñido, J. D., & Portas, S. O. (2019). FILIPINO BASED FACIAL EMOTION FEATURES DATASETS USING HAAR-CASCADE CLASSIFIER AND FISHERFACES LINEAR DISCRIMINANT ANALYSIS ALGORITHM. https://ejournals.ph/article.php?id=14064

Prospero, M. R., & Garcia, P. C. (2020). Asynchronous Facial Emotion Recognition (FER) of the faculty members of College of Engineering and Computer. https://ejournals.ph/article.php?id=17406

Yong, E. D., De Jesus, L. C. M., Brucal, S. G. E., Samaniego, L. A., Jr, Peruda, S. R., Jr, Cosio, P., Villarroel, J. M., Novilla, K., Cacalda, E. a. M., Lacar, C. B., & Calibara, A. E. (2023, April 13). RAMY Greeting Feature using HAAR cascade classifier and HOG Algorithm for Asia Pacific College. https://stepacademic.net/ijcsr/article/view/406

GRADING SHEET

Thesis Title	**A Modified Haar Cascade Algorithm for Enhanced Facial Recognition Applied in PLM Violation System**
Name of Students	**Antipona, Clarence A.**
	Magsino III, Romeo R.

Rating

10 - 8 Outstanding **7-4 Good** **1-3 Fair**

Criteria	Outstanding	Good	Fair
Academic Significance		6	
Contribution to the Community	8		
Technical Novelty		7	
Quality of Information	9		
Knowledge of the Student to the Topic	8		
Antipona, Clarence A.	8		
Magsino III, Romeo R.	8		
		Total Rating	

Panel's Recommendation:

- Revise the SOP and make it clear, state the effects of each problem instead of just stating the problem alone.
- Revise the Objective and make it focus in answering each SOP, state the process on how the Objective will be done.
- Remove the Result to the Methodology because it's not part of the Chapter 3

John Ray M. Tenio
Name & Signature of Panel

GRADING SHEET

Thesis Title	**A Modified Haar Cascade Algorithm for Enhanced Facial Recognition Applied in PLM Violation System**
Name of Students	**Antipona, Clarence A.**
	Magsino III, Romeo R.

Rating

10 - 8 Outstanding **7-4 Good** **1-3 Fair**

Criteria	**Outstanding**	**Good**	**Fair**
Academic Significance			
Contribution to the Community			
Technical Novelty			
Quality of Information			
Knowledge of the Student to the Topic			
Antipona, Clarence A.			
Magsino III, Romeo R.			
		Total Rating	

Panel's Recommendation:

- Revise the SOP and make it clear, state the effects of each problem instead of just stating the problem alone.
- Revise the Objective and make it focus in answering each SOP, state the process on how the Objective will be done.
- Remove the Result to the Methodology because it's not part of the Chapter 3

Khatalyn E. Mata
Name & Signature of Panel