Enhancement of Real-Time Face Recognition via Video Surveillance System

Lucky Lhaura Van FC[1*], Noppharat Hakam Singh[2], Sharifah Saon[2], Abd Kadir Mahamad[2]

[1]Department of Informatics Engineering, Faculty of Computer Science
University of Lancang Kuning, Pekanbaru, INDONESIA

[2]Faculty of Electrical & Electronic Engineering,
Universiti Tun Hussein Onn Malaysia, MALAYSIA

*Corresponding Author

Abstract:
This project report is about the investigation to enhance the accuracy of the face recognition video surveillance system. Recently, video surveillance with Closed Circuit TV (CCTV) cameras to analyze crowd people activities and real-time face recognition have been widely used. There are many face recognition systems that use Haar Cascade feature, through a classification method called AdaBoost. However, due to the limitations of Haar Cascade, there are many methods and algorithms that have been implemented to enhance the accuracy of face recognition. One of them is the Gray Level Co-occurrence Matrix (GLCM), which analyse image features through homogeneity and correlation. The hardware and software used in this project are Raspberry Pi, Pi camera, OpenCV library, and Scikit-Image library. The developed system consists of four phases for the operating method, which are data gathering, GLCM image texturizing, train recognizer, and recognition. Thonny Python was used in all of these phases with the OpenCV library and Scikit-Image library applied. All the results of this project were based on face detection, face recognition with face complications, face recognition with distances, and face recognition with different face angles. The results confirmed that the system was reliable with human face detection and face recognition with an accuracy up to 68%. Future research could explore optimizing performance under challenging conditions, like varying light and angles, or deploying more powerful hardware for better efficiency and application such as building access.

Keywords:
Surveillance camera, Face recognition, Raspberry Pi, Haar cascade classifier, Gray level co-occurrence matrix

1. Introduction

Nowadays, there have been countless security and surveillance systems used for monitoring and observing surroundings for safety purposes. These types of security systems are also often used to detect moving objects by using surveillance electronic equipment such as authentication systems, robotics systems, and video surveillance [1]. Therefore in recent years, video surveillance uses Closed

*Corresponding author: luckylhaura90@gmail.com

Circuit TV (CCTV) camera to analyze the crowd people activities [2]. Meanwhile, due to the limitation of traditional CCTV which has low resolution, there has been much research done by the researcher to improve the quality of the camera and security.

One CCTV camera feature that has been improved is real-time face recognition that detects faces with better system performance and speed optimization [3]. Real-time face recognition can be applied in several areas such as building entry, highway traffic, airports, and terminals [1]. There has estimated that more than 770 million surveillance cameras already installed in 2020 and will have a total of 1 billion cameras in 2021 with more than half will be Asia.

In order to improve face recognition, some of the features have been applied which as the Haar Cascade. Haar cascade were proposed by Paul Viola Michael Jones that uses Haar features based on the Adaboost cascade [4]. Face detection by using Haar feature measures the face using the value of pixels (x,y) at which x stands for width and y stands for height of the face [5]. Viola Jones method is better than Haar Cascade but it also has its flaws such as the hardship of locating facial features because of several weaknesses (illumination, noise, occlusion) and difficulty in detecting features because of complex background [6].

Face recognition is the ability of a computer to recognize a human face from an image or a live video [7]. Over the last decade, face recognition has become an important field of study in a number of areas such as computer vision, neuroscience, pattern recognition, psychology, machine learning, and image processing [8]. The face recognition system has an input image in database that able to search the person's identification. The face recognition program consists of four primary methods, which are identification, alignment, extraction, and standardization.

In the case of video, face detection tracking is needed to detect a face while it is in motion [9]. The alignment of the face aims to attain a more precise position and normalization of the face, while the identification of the face eliminates rough estimates of the size and position of the face. These facial elements are the mouth, nose, eyes, and other facial contours. The input face image is transformed with respect to geometrical properties such as pose and scale by geometric transformation or morphing [9] [10]. Then, the extraction function is performed to provide objective data that are useful for distinguishing between different people's faces and stability with respect to geometric and photometric variations [9]. Finally, the matching face extracts the vector of the input face in the database to match the faces where the device tells who the person identified and checks to guess the identity and tell whether it is a true or false guess [7] [11].

"Video" refers to motion pictures in digital or analog form [12]. Video surveillance is becoming an increasingly effective tool in organizations that aim to secure physical and capital properties and in the same time, to monitor more individuals, places, and activities with a willingness to gain more useful information [13]. Gaussian Mix Model (GMM) is one of the quick techniques of extracting frames from a different but the most correct and complex algorithm for extracting moving objects in a video sequence [14]. GMM is a common basis for the computer vision subtraction method due to its robustness to subtle changes in illumination. The bottleneck is the computational intensity due to the need to calculate and update GMM as a substitute for using a fixed number of GMM [15]. The reason for using a robust surveillance system specifically is because it aims at a low value of false positive error [16] due to surveillance guards can deviate from too many alarms triggered by a moving rain, small camera motion, tress, or varying lighting conditions.

The two algorithms that will be used in this system are Haar Cascade and Gray Level Co-occurrence Matrix (GLCM). Haar Cascade is a computer that learns to detect objects in a video or an image [9] using these four methods, which are Haar-like features, integral image, AdaBoost learning, and Cascade

Classifier [16]-[19]. Due to it does not rely on any pixel value of an image, the Haar-like feature offers fast computing where the Haar-like feature value was calculated using an integral image that could calculate the value precisely and quickly by generating a new image presentation using the value of the earlier Haar-like feature scanned. The *Eq.* (1) and (2) is used to measure the number of pixels in the rectangular. The pixel at (x,y) is the value of the i(x,y) in *Eq.* (1) while the I(x,y) is the sum of an integral of pixel values in *Eq.* (2).

$$I(x,y) = \sum_{\substack{x'<x \\ y'<y}} i(x',y') \quad \textit{Eq. 1}$$

$$I(x,y) = i(x,y) + I(x,y-1) + I(x-1,y) + I(x-1,y-1) \quad \textit{Eq. 2}$$

The purpose of GLCM is to measure the heterogeneous surface pattern and roughness shown in digital images, where each index is highlighted with a certain texture property [20][21]. Texture is the term used to characterize the tonal or gray-level variations in an image. The gray-scale image in the first matrix was first subdivided into an integer matrix by dividing the continuous pixel value range into Ng equal width bins also known as the number gray-level, after the bin values were mapped into a single gray-level. The GLCM elements are then determined based on how often pairs of pixels with a given gray level and a specified spatial relationship occur in the matrix. The four different angles 0º, 45º, 90º, and 135º are known as pixel adjacency. After that, the GLCM is normalized to calculate the sum of all the elements that will be equal to one later. Lastly, it is possible to calculate the number of gray levels and the normalized matrix elements [19]. There are a few important features such as Angular Second Moment (ASM), Inverse Difference Moment (IDM), Entropy, Dissimilarity, and Correlation [20] [21].

ASM is the sum of squares of entries in the GLCM which is used to measure the image homogeneity. From *Eq.* (3), the ASM will be high when the pixels are very similar or the image has a good homogeneity. The i,j are the spatial coordinates of the function p(i,j) and Ng is gray level. More specifically i was the probability of grey level and j was the neighboring of grey level. IDM is the local homogeneity where it will be high when the gray level is uniform and the inverse GLCM is high. *Eq.* (4) formulates the IDM. The Entropy shows the information of the image that needed to be compressed. The *Eq.* (5) shows the Entropy compression of an image. Lastly, the Correlation measured the linear dependency of gray levels of neighboring pixels as shown in *Eq.* (6).

$$ASM = \sum_{i=0}^{Ng-1} \sum_{j=0}^{Ng-1} P_{i,j}^2 \quad \textit{Eq. 3}$$

$$IDM = \frac{\sum_{i=0}^{Ng} \sum_{j=0}^{Ng} P_{ij}}{1+(i-j)^2} \quad \textit{Eq. 4}$$

$$Entropy = \sum_{i=0}^{Ng-1} \sum_{j=0}^{Ng} -P_{ij}(logP_{ij}) \quad \textit{Eq. 5}$$

$$Correlation = \sum_{i,j=0}^{N-1} P_{i,j} \left[\frac{(i-\mu_i)(j-\mu_j)}{\sqrt{(\sigma_i^2)(\sigma_j^2)}} \right] \quad \textit{Eq. 6}$$

2. Project Development

Figure 1 shows the project flowchart for project development. The project starts by gathering data, usually in the photo form of the person to be identified. Such data are then stored in a folder called 'dataset'. After the data is processed, the data will be textured using two of the GLCM texture properties, which are dissimilarity and correlation. The system would then be trained based on the data from the dataset folder. Finally, the identification of a person's face depends on the data gathered, whether it is recognized or otherwise. In the flowchart, a threshold value is used to decide if the detected face data is confident enough to be considered a match. This value determines the system's sensitivity: higher

thresholds reduce false matches but may miss true ones, while lower thresholds increase matches but risk more false positives. The flowchart outlines the steps in developing the face recognition system.

(i) **Data Gathering**: Collect face images, which are stored in a dataset.
(ii) **GLCM Image Texturizing**: Analyze textures in the images to create unique features using GLCM properties.
(iii) **Training the Recognizer**: Use collected data to train a recognizer, creating a model that can identify faces.
(iv) **Recognition**: Match detected faces against the trained model to identify individuals.

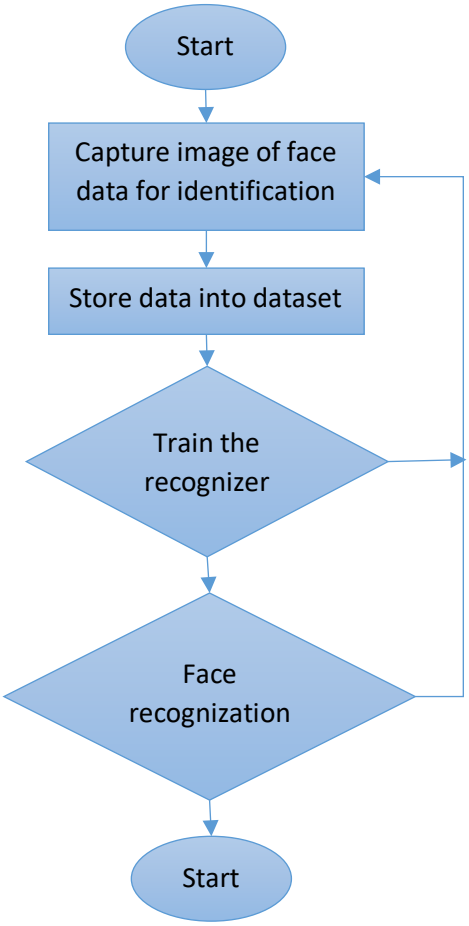


Figure 1: Project flowchart

2.1 Project Hardware

The section is about all the hardware and software used in this project. The hardware consists of the Pi camera module and the Raspberry Pi board. The software that will used to program the system is Thonny Python IDE. Raspberry Pi Camera Module Rev 1.3 has 5 Megapixels of lens that provide a 720p resolution video at 60 frames per second and 1080p video resolution at 30 frames per second. The Raspberry Pi model used in this project is Raspberry Pi 3 Model B. The reasons this model has been used are its versatility in size, low power consumption, low price for a single-board device, WiFi connectivity, Bluetooth connectivity, camera port, and SD card that allows up to 128 GB of OS storage and data.

Thonny is a free Python Integrated Development Environment (IDE) specifically designed for beginners and it was used to write and compile the code. OpenCV is a free computer vision that can manipulate images and videos to perform different tasks, from viewing a live webcam or camera feed

to training a robot to recognize a face. In this project, some of the OpenCV libraries were used to identify face features such as the eye and the front face. Scikit-Image is an open-source compilation of an image-processing algorithm library primarily for Python programming language, Cython programming language, and C programming languages. The scikit-Image algorithm involves transformations, color space manipulation, visualization, filtering, and more. It is designed to operate in tandem with the NumPy numerical, SciPy numerical, and scientific Python libraries.

2.2 Data Collection

Face detection is the most important function of facial recognition. The Haar Cascade Classifier is the most common way to detect a face and the pre-trained classifier is already included in the OpenCV directory file. In this project, only two of the pre-trained Haar Cascade Classifiers were used, which are called haarcascade_frontalface_default.xml for facial detection and haarcascade_eye.xml for eye detection. The minimum requirement for the data samples is 30 but for this project purpose, the data samples gathered is 200 as the lower amount of data, the accuracy is also lower. The data gathering coding will run and the Pi camera will record the face samples, gathered and stored in a database folder called 'dataset'. That is how the program classifies the individual to be recognized.

2.3 GLCM Image Texturizing

Once the data is gathered, the data will be texturized by using two of the GLCM properties from the Scikit-Image library, which are the greycomatrix and greycoprops. The greycomatrix purpose is to calculate the GLCM based on the image, distance, angles, levels, symmetric, and normed. The image is one of the samples in which the texture will be extracted from the background. The distances are the list of pixel pair distance offsets, the angles are the list of pixel angles in radians, and the levels are also known as a number of gray level, Ng are the integers in an input image which was 256 for an 8-bit image. The symmetric would be 'True' because the output matrix becomes symmetric by ignoring the value pair order, such that both (i, j) and (j, i) are cumulative because (i, j) becomes encountered for a given offset. As for the normed when 'True', it will normalize each matrix by splitting the cumulative number of aggregate co-occurrences for the offset. Elements of the corresponding total of the equation to 1.

The greycoprops aimed to measure the texture properties of the GLCM by computing one or more GLCM properties to serve as a concise description of the matrix. The properties computed are dissimilarity and correlation. The texture of the correlation tests the linear dependency of the grey rates on certain surrounding pixels and the texture of the dissimilarity reflects the pair of similarities between the square and the symmetric. These are the properties that were used for texturizing at the background of the image. The background locations in x-axis and y-axis were identified by using the paint application and those locations are [(207, 20), (193, 79), (206, 190), (21, 191), (6, 47)].

2.4 Train Recognizer

Next, the system needs to be trained every time new data is updated to allow the system to recognize it. Each data gathered was assigned with an ID so that the system recognizes a different ID for different person. A new code must be run for the train recognizer to function, which will create a file named trainer.yml in a folder called the trainer. The yml file is essential for face recognition so the system is able to recognize the dataset. The system will predict the identity of the face detected face based on the ID in the dataset.

2.5 Face Recognition

The last phase of the project is the recognition of the face where the system can recognize the face based on the front face and the eye. A new set of codes will be run for the face recognition to operate. Figure 2 shows the face recognition for Noppharat_DE160045 is the same as the ID stored in the dataset. The system will match based on front-face detection and eye detection.

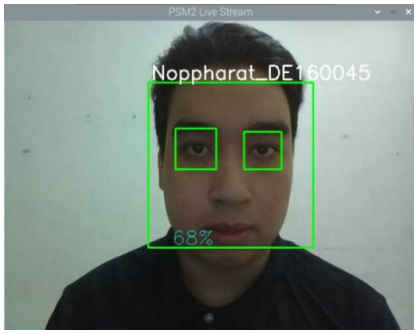


Figure 2: System recognized Noppharat_DE160045

3. Results and Discussion

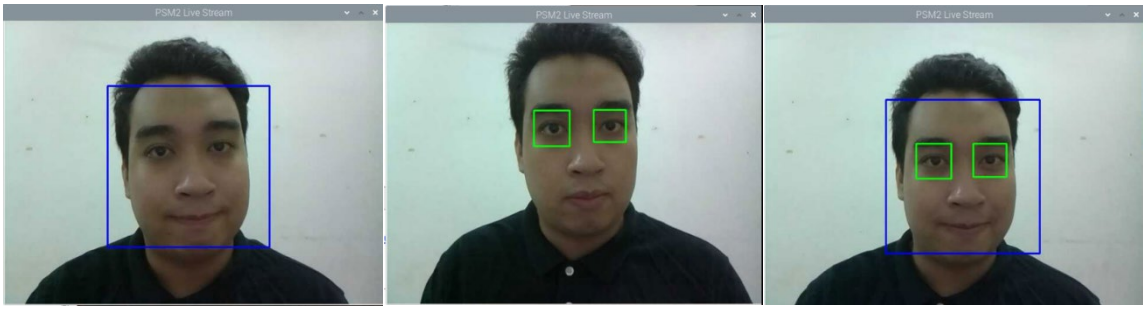
This section discusses the result and overall analysis of the Enhancement of Real-time Face Recognition via Video Surveillance System. All the results of this project that vary from the accuracy of face complication to the accuracy of face angle are stated in the section. These results are collected based on the simulations that run with Raspberry Pi and Pi cameras. This section also included a comparison with the previous project Real Time Recognition of Video Surveillance System [25].

3.1 Face Detection

Once project development was completed, the first result obtained was the face detection with the implementation from two of the Haar Cascade Classifiers by the system once the system was activated and displayed a continuous real-time live video. The first detection was the detection of the front face where the blue color rectangular box was drawn by the system as shown in (a) of Figure 3. Next is eye detection where two green color rectangular boxes were drawn by the system around both eyes as shown in (b) of Figure 3. Lastly, both front face and eyes detection were detected automatically the both front face and eyes of a person as shown in (c) of Figure 3.

Table 1: Face detection

No	Detection Part
1	Front face
2	Eyes
3	Front face and eyes



(a) **(b)** **(c)**

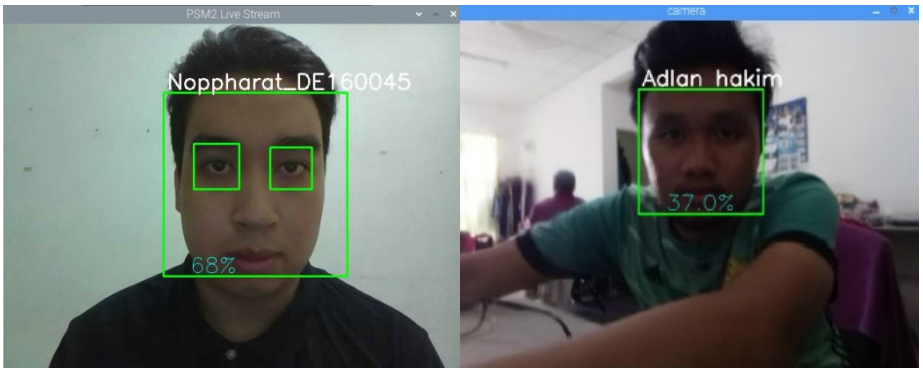
Figure 3. Face detection of (a) front face, (b) eyes, and (c) front face and eyes

3.2 Face Recognition

By using two of the Haar Cascade Classifiers, the system produced results of recognition of human faces, as in Figure 4. With the GLCM image texturizing added to the system, the accuracy of face recognition was even higher. Table 2 shows the comparison between the current face recognition project that has front face detection, eye detection, and GLCM image texturizing (a) with the previous face recognition project (b) that has only used front face detection. The accuracy of face recognition in the current project is at 68% accuracy compared with the previous project which is only at 37%. The result also displayed the name of the person based on the ID settings in a line of code.

Table 2: Face detection

No	Algorithm	Accuracy
1	Face recognition using GLCM	68%
2	Face recognition using Haar Cascade	37%



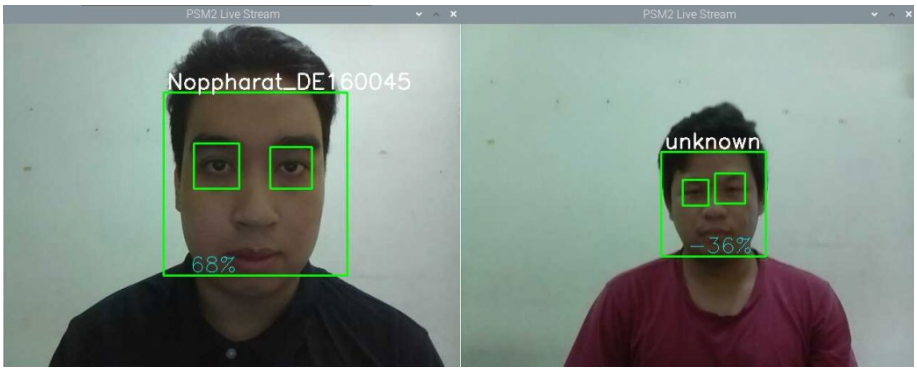
(a) **(b)**

Figure 4: Comparison GLCMs (a) and Haar cascade (b) Face recognition project

The results indicate the name of the person based on the ID of the user settings in the code. If the ID of a user is not in the database records, the program won't recognize that person, and an unknown record is shown. The Front face detection and eye detection are shown regardless the system recognized or didn't recognize that person. In Table 3, the sample shows the difference between the face that was recognized (a) and the face that was unrecognized (b) by the system as shown in Figure 5. In order to ensure that the system was reliable, a few tests were carried out to test the reliability of the face recognition system. Such tests consist of complications of the face, distances of the face, and the angles of the face.

Table 3: Face recognition

No	Algorithm	Accuracy
1	Face recognized	68%
2	Face unrecognized	-36%



(a) **(b)**

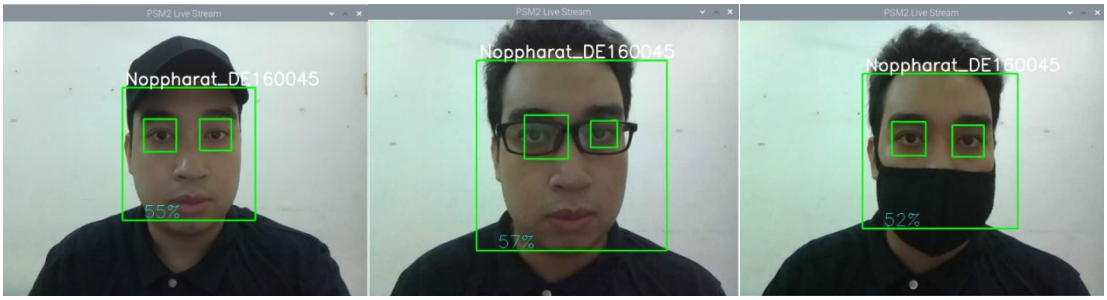
Figure 5: Face recognition

3.3 Face Recognition with Face Complications

Face complications of face abnormalities were one of the tests conducted to test the accuracy of the system. This test focused on three parts of the face which are head, eyes, and mouth. The part of the head was covered by a cap (a), the part of the eyes covered with glasses (b), and part of mouth was covered with a face mask (c) as shown in Figure 6. Each of these parts was covered to test if the system was still able to recognize the face. Table 4 shows the face recognition with face complications. As more parts of the face were covered, the accuracy of recognition dropped as shown in Figure 7. The lighting and the color of the background also play a part in the recognition of the face as the GLCM image texturizing in the system texturizes the background part of image data in the database.

Table 4: Result of face with complications

No	Covered Part	Recognition	Accuracy
1	Mouth	Yes	52%
2	Head	Yes	55%
3	Eye	Yes	57%
4	Mouth and eye	Yes	49%
5	Mouth and head	Yes	48%
6	Head and eye	Yes	52%
7	Mouth, eye, and head	Yes	43%



(a) **(b)** **(c)**

Figure 6: Head covered (a), eye covered (b), and mouth covered (c)

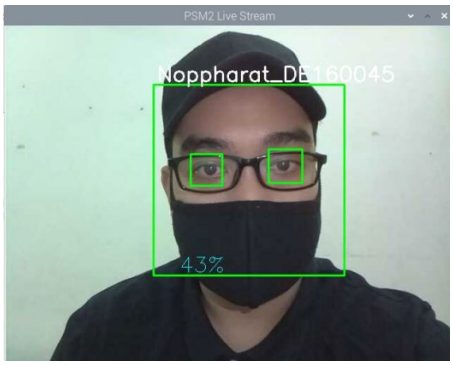


Figure 7: Head, eye, and mouth covered

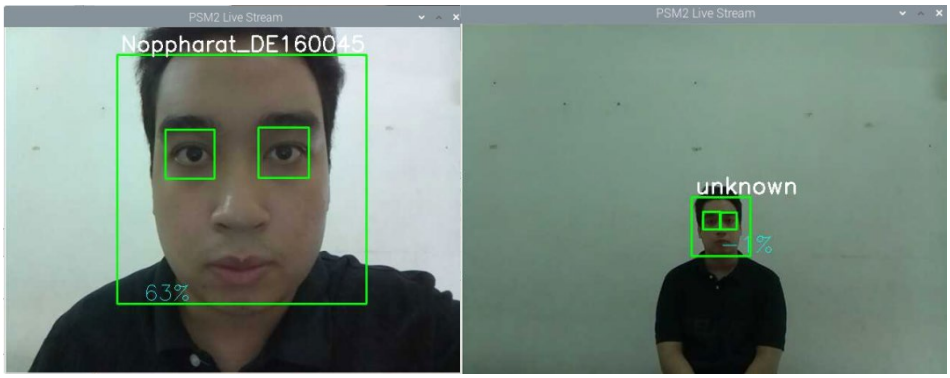
3.4 Face Recognition with Distance

This system also has a distance limit that can be processed for the recognition of the face. The further the face between the Pi camera, the lower the accuracy obtained. Table 5 shows the distance tested for face recognition samples. The measurement has been measured until the face is no longer known or recognized by the system. The recognition accuracy dropped as the distance between the face and the camera increased. The nearest distance possible between the face and the camera was estimated at 0.3m (a). If the distance is over 1.3m (b), the system won't recognize the face but face detection and eye detection still work as shown in Figure 8.

Typically, face recognition systems perform best within a specific range that balances clarity and system efficiency. For instance, distances between 0.3m to 1.5m are commonly tested to simulate realistic surveillance scenarios. However, for professional-grade systems, standardized testing protocols, such as those recommended by organizations like the National Institute of Standards and Technology (NIST), ensure consistency and comparability across different face recognition systems. Incorporating such standards into testing can help determine the reliability of the system under common usage conditions.

Table 5: Result of face with complications

No	Estimate Distance	Recognition	Accuracy
1	0.3m	Yes	63%
2	0.5m	Yes	58%
3	0.8m	Yes	42%
4	1.0m	Yes	34%
5	1.3m	Yes	19%
6	1.5m	No	-1%



(a) **(b)**

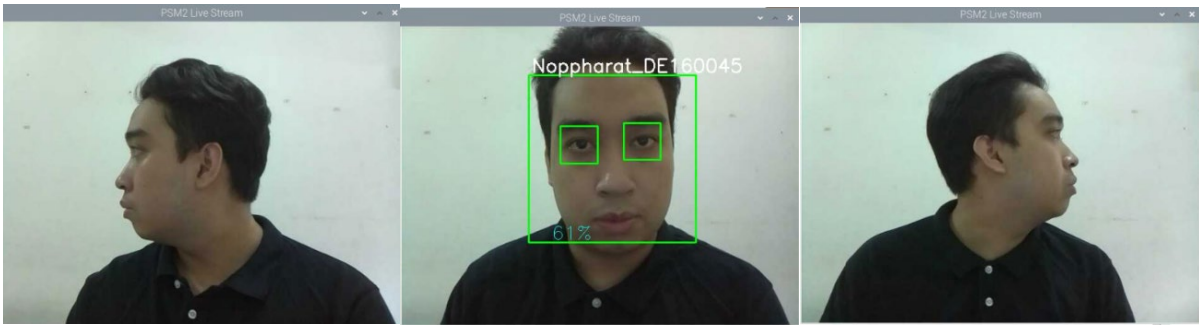
Figure 8: Estimate distance at 0.3m (a) and estimate distance at 1.3m (b)

3.5 Face Recognition with Face Angle

Face recognition systems nowadays even recognize a person from any angle, especially the machine that uses the Adaboost classifier algorithm that is able to detect multi-angle humans. For the system that used Haar Cascade classifier, the face recognition also could recognize the face based on the angle of the faces. Table 6 shows the results of the face recognition test with several face angles and Figure 9 shows the estimated angle at -90º (a), 0º (b), and 90º (c).

Table 6: Result of face recognition test with face angle

No	Estimate Angle	Recognition	Accuracy
1	90º	No	0%
2	65º	No	0%
3	45º	Yes	53%
4	25º	Yes	57%
5	0º	Yes	61%
6	-25º	Yes	53%
7	-45º	Yes	41%
8	-65º	No	0%
9	-90º	No	0%



(a) **(b)** **(c)**

Figure 8: Face angle at -90º (a), 0º (b) and 90º

4. Discussion

The quality of the images significantly affects the performance of face recognition systems. Key factors include:

1. **Resolution**: Higher-resolution images provide more detail, allowing the system to identify facial features accurately. Low-resolution images, especially in low-light conditions, can lead to reduced accuracy.
2. **White Balance**: Proper white balance ensures that colors are represented accurately. Inconsistent or incorrect white balance can distort the appearance of skin tones and facial features, impacting recognition reliability.
3. **Bit Depth**: The bit depth of an image refers to the amount of information stored for each pixel. A higher bit depth, such as 24-bit color images, offers better color fidelity and more nuanced details, which are critical for distinguishing subtle facial features.
4. **Aspect Ratio**: The aspect ratio determines the proportions of the image. Maintaining a correct aspect ratio ensures that the image does not appear stretched or distorted, which is crucial for accurate facial analysis.

5. Conclusion

At the end of this project, all the objectives for this study have been completed by using hardware and software such as the Raspberry Pi, Pi camera, OpenCV library, and Scikit-Image library. For the OpenCV library, the Haar Cascade classifiers used are the front face and eyes while the Scikit-Image library used the GLCM parameter and GLCM properties. Several tests have been conducted and compared for the face detection and face recognition. The results confirmed that the system was reliable with human face detection and face recognition. However, there are some limitations found in this study. These limitations are the face recognition system during a live camera feed used in Raspberry Pi has a few milliseconds delay. This is because the Raspberry Pi 3 Model B has low performance in terms of processor and memory speed. Other limitations are the light conditions around the environment because the system does not run at its best in low-light conditions. Overall from this project, the Enhancement of Real-time Face Recognition via Video Surveillance System was better (68% accuracy) than the previous system that did not use the GLCM image texturizing process (37%).

Acknowledgments

This research was supported by the Universiti Tun Hussein Onn Malaysia (UTHM)

References

[1] E. Omar, A. Noor and S. Somaya, "A Review of Video Surveillance Systems," Journal of Visual Communication and Image Representation, vol. 77, 2021, https://doi.org/10.1016/j.jvcir.2021.103116

[2] E.L. Piza, "CCTV VIDEO SURVEILLANCE AND CRIME CONTROL." The Oxford Handbook of Evidence-Based Crime and Justice Policy (2024): pp. 265.

[3] G. Singh and A. K. Goel, "Face Detection and Recognition System using Digital Image Processing," *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, Bangalore, India, 2020, pp. 348-352, doi: 10.1109/ICIMIA48430.2020.9074838.

[4] M. S. Minu, K. Arun, A. Tiwari, and P.Rampuria, "Face recognition system based on haar cascade classifier," International Journal of Advanced Science and Technology, vol. 29, no. 5, pp. 3799-3805, 2020.

[5] H. -T. Ho, L. Vuong Nguyen, T. Huong Thi Le and O. -J. Lee, "Face Detection Using Eigenfaces: A Comprehensive Review," in *IEEE Access*, vol. 12, pp. 118406-118426, 2024, doi: 10.1109/ACCESS.2024.3435964.

[6] J. M. Al-Tuwaijari and S. A. Shaker, "Face Detection System Based Viola-Jones Algorithm," *2020 6th International Engineering Conference "Sustainable Technology and Development" (IEC)*, Erbil, Iraq, 2020, pp. 211-215, doi: 10.1109/IEC49899.2020.9122927.

[7] J. H. Joe Minichino, "Learning OpenCV 3 Computer Vision with Python," vol. II, pp. 181-182, 2015.

[8] Adjabi, I.; Ouahabi, A.; Benzaoui, A.; Taleb-Ahmed, A. "Past, Present, and Future of Face Recognition: A Review," *Electronics* 2020, *9*, 1188. https://doi.org/10.3390/electronics9081188

[9] T. Le, "Applying Artificial Neural Networks for Face Recognition," Advances in Artificial Neural Systems, pp. 1-2, 2011.

[10] S. Li and W. Deng, "Deep Facial Expression Recognition: A Survey," in *IEEE Transactions on Affective Computing*, vol. 13, no. 3, pp. 1195-1215, 1 July-Sept. 2022, doi: 10.1109/TAFFC.2020.2981446.

[11] A. Divya, K.B. Raja and K.R. Venugopal, "Windowing approach for face recognition using the spatial-temporal method and artificial neural network," vo. 6, no. 2, pp. 124-162, 2022, https://doi.org/10.1504/IJAPR.2020.111513.

[12] J. Xu, "A deep learning approach to building an intelligent video surveillance system," *Multimed Tools Appl*, vol. 80, pp. 5495–5515, 2021, https://doi.org/10.1007/s11042-020-09964-6.

[13] C.S. Sung, J.Y. Park, "Design of an intelligent video surveillance system for crime prevention: applying deep learning technology," *Multimed Tools Appl*, vol. 80, pp. 34297–34309, 2021, https://doi.org/10.1007/s11042-021-10809-z.

[14] M. Purohit, M. Singh, S. Yadav, A. K. Singh, A. Kumar and B. K. Kaushik, "Multi-Sensor Surveillance System Based on Integrated Video Analytics," in *IEEE Sensors Journal*, vol. 22, no. 11, pp. 10207-10222, 1 June1, 2022, doi: 10.1109/JSEN.2021.3131579..

[15] C. S. a. W. E. L. Grimson, "Adaptive Background Mixture Models for Real-Time Tracking," Computer Vision and Pattern Recognition Fort Collins, pp. 246-252, 1999.

[16] Y. H. a. R. O. H. T. A. Ezzahout, "Detection evaluation and testing region incoming people's in a simple camera view," International Conference on Inovative Computing Technologies, 2012.

[17] D.N., C., G., A., M., R. (2012). Face Detection Using a Boosted Cascade of Features Using OpenCV. In: Venugopal, K.R., Patnaik, L.M. (eds) Wireless Networks and Computational Intelligence. ICIP 2012. Communications in Computer and Information Science, vol 292. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-31686-9_46.

[18] C. G. B. P. W. a. M. A. S. D. Tyas Purwa Hapsari, "Face Detection Using Haar Cascade in Difference Illumination," Proc. - 2018 Int. Semin. Appl. Technol. Inf. Commun. Creat. Technol. Hum. Life, iSemantic, pp. 555-559, 2018.

[19] A. M. a. K. Abid, "OpenCV-Python Tutorials Documentation," OpenCV Python Doc, pp. 1-269, 2017.

[20] D. P. a. K. Soroka, "Research of usage of Haar-like features and AdaBoost algorithm in Viola-Jones method of object detection," 2013 12th International Conference on the Experience of Designing and Application of CAD Systems in Microelectronics (CADSM), pp. 284-286, 2013.

[21] K. S. a. I. D. R.E. Haralick, "Textural Features for Image Classification," IEEE Transactions on Systems, Man and Cybernetics, vol. 3(6), pp. pp 610-621, 1973.

[22] S. K. B. A. K. Rahul, "Face Recognition By Linear Discriminant Analysis," International Journal Of Communication Network Security, vol. 2(2), 2013.

[23] Vera, A. Kusnadi, I. Z. Pane, M. V. Overbeek and S. G. Prasetya, "Face Recognition Accuracy Improving Using Gray Level Co-occurrence Matrix Selection Feature Algorithm," *2023 International Conference on Smart Computing and Application (ICSCA)*, Hail, Saudi Arabia, 2023, pp. 1-6, doi: 10.1109/ICSCA57840.2023.10087414.

[24] A. P. a. M. Habibi, "Face Detection using Haar Cascades to Filter Selfie Face Image on Instagram," 2019 International Conference of Artificial Intelligence and Information Technology (ICAIIT), pp. 6-9, 2019.

[25] Ahmad, A. H., Saon, S., Abd Kadir Mahamad, C. D., Wiwoho, S., Mudjanarko, S. M. S. N., & Hariadi, M. (2021). Real time face recognition of video surveillance system using haar cascade classifier. Indonesian Journal of Electrical Engineering and Computer Science, 21(3), 1389-1399. http://doi.org/10.11591/ijeecs.v21.i3.pp1389-1399