Attendance System based on Face Recognition, Face Mask and Body Temperature Detection on Raspberry Pi

2021 International Seminar on Application for Technology of Information and Communication (iSemantic) | 978-1-6654-2804-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/ISEMANTIC52711.2021.9573182

Handayani Saptaji Winahyu
Master in Applied Telecommunication Engineering Politeknik Negeri Semarang
Semarang, Indonesia
handayani.mtr19@polines.ac.id

Eni Dwi Wardihani
Master in Applied Telecommunication Engineering Politeknik Negeri Semarang
Semarang, Indonesia
edwardihani@polines.ac.id

Samuel Beta
Master in Applied Telecommunication Engineering Politeknik Negeri Semarang
Semarang, Indonesia
sambetak2 @polines.ac.id

Abstract — **The world is currently experiencing a COVID-19 pandemic which has caused many deaths. Then, one of COVID-19 early detection can be done through wearing a face mask and detecting body temperature when entering a room. The purpose of this research is to create an attendance system capable to recognize face, face mask, and measure body temperature. The data was tested against three types of Raspberry Pi (Pi 3B, Pi 4-4Gb, and Pi 4-8Gb). Each type of Raspberry Pi was tested using three metric combinations of different encoding methods and object classification, namely Method 1 - Haar Cascade and LBPH, Method 2 - Haar Cascade, and Tensorflow, Method 3 - MTCNN and Tensorflow. The test results obtained Method 1: FPS 15, accuracy rate 60%, CPU temperature 58^{0}C, Method 2: FPS 4, accuracy rate 90%, CPU temperature 65^{0}C, Method 3: FPS 2, accuracy rate 95%, CPU temperature 68^{0}C. Based on the benchmarks and scoring, it can be concluded that the most optimal attendance system based on face recognition, mask, and body temperature detection is a system with a Raspberry Pi 4–4Gb and Method 1-Haar Cascade and LBPH. For next development, this attendance system needs to be further improved in security and accuracy without increasing CPU load.**

Keywords—artificial intelligence, attendance system, body temperature, face recognition, face mask, Raspberry Pi

I. PENDAHULUAN

Face is one of the biometric parameters that can be used as an identifier in addition to fingerprints, palms, voice, iris and retina.[1] The advantage of using biometric parameters as identifiers are a unique key and hard to be falsified. Departing from this, biometric-based identifiers are widely used for attendance systems and door access, both at the research level[2][3] and commercial products[4]. The most popular is fingerprints as biometric identifier for an access system. However, there are still drawbacks to using this technique because the sensor is less accurate in identifying under certain conditions (injured/wet/sweatened fingers). The sensor also must be cleaned frequently due to dust/dirt attached due to frequent use. Other drawback, fingerprints can be duplicated with certain techniques[5]. Therefore, an alternative non-contact bbiometric identifier is needed, which does not come into direct contact with the sensor.

Face recognition is a biometric identification technique that does not require physical contact with objects. There are two stages in facial recognition computing, namely face detection (encoding) and prediction/identification based on face classification. Some of the facial recognition encoding methods that are often used include: Haar Cascade[6], DLIB[7], and MTCNN[8]. The Haar Cascade method is carried out by detecting facial features according to the Viola-Jones algorithm, the Dlib method is conducted by detecting 68 points of facial landmarks, while the MTCNN (Multi Task Cascaded Convolutional Neural Network) method detects faces through 5 face landmark points based on the CNN algorithm.

For facial prediction/classification methods that are often used include the LBPH (Local Binary Pattern Histogram) algorithm, Caffe, Panda, Tensorflow and Keras. The LBPH algorithm has the advantages of high confidence, minimum noise and high efficiency[9], while other algorithms are based on machine learning which requires high computational resources[8][10].

Several previous researches have investigated facial recognition using Python and OpenCV software[11][12], and implemented for door access system using Raspberry Pi [13][14]. However, no one has researched facial recognition when using a medical mask. Previous research has also not benchmarked several methods to test the Raspberry Pi can optimally computes the facial recognition so far.

Departing from this and as a novelty in this research, three types of Raspberry Pi and three facial recognition methods were tested to find the most optimal facial recognition system (and masks) as a technology base for attendance systems that use limited hardware resources such as the Raspberry Pi..

II. RESEARCH METHOD

The methods used in this study was implemented by testing three types of Raspberry Pi, namely Raspberry Pi 3B, Raspberry Pi 4 with 4Gb RAM and Raspberry 4 with 8B RAM. Each type was then tested with 3 main methods of heuristic facial recognition to obtain Raspberry Pi performance data in the computing process. The method chosen is a method that can represent the level of accuracy and utilization of hardware resources used to run it. The first method chosen is:

1) *Haar Cascade dan LBPH Method.* Haar Cascade as a face detection method based on features, does not include the realm of machine learning so it does not use much memory resources. This method is

combined with the LBPH classification algorithm which does not belong to deep learning category.[15]

2) *Haar Cascade and Tensorflow Method*. Just like the first method, but the tensorflow library which is included as deep learning algorithm. The expected objective by using the second method compared to the first method is to be able to find out the difference in the use of deep learning algorithms (represented by tensorflow) using the same face detection method (Haar Cascade).

3) *Metode MTCNN dan Tensorflow*. The MTCNN method uses an algorithm that is included in the machine learning category, so that the selection of this method already represents the level of use of machine learning/deep learning in face recognition compared to the two previous methods..

Meanwhile, the parameters tested are:

1) *Frame per second (FPS).* This parameter was chosen as an indicator of the CPU's ability to perform computing processing[16], especially in scripting facial recognition algorithms.

2) *CPU Temperature*. This parameter also shows an indication of the CPU's ability to process the scripts assigned to it.

3) *Accuration rate*. The accuracy of each method in recognizing faces is measured by the level of confidence to determine the performance of each method when run on the Raspberry Pi.

From the three stratified methods above and applied to different hardware resources (three types of Raspberry Pi) and different parameters, it is hoped that valid testing metrics (benchmarks) will be obtained to accurate final research results.

A. System Design

The design of an attendance system based on face recognition, masks and body temperature based on the Raspberry Pi can be shown in Figure 1.

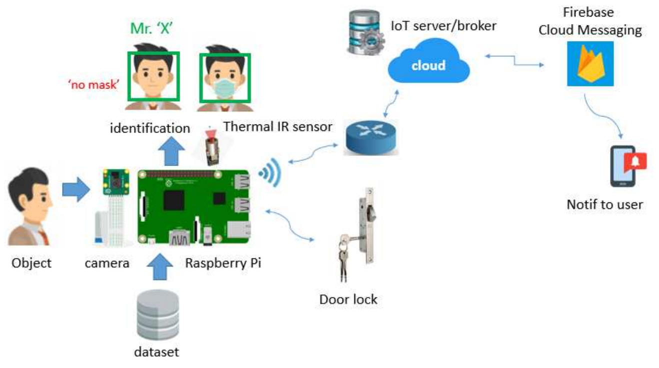

Figure 1. Attendance system design based on facial recognition

In the figure, it can be seen that the face, mask and body temperature recognition process is carried out after the object faces the camera. The Raspberry Pi then takes an image through the camera, detecting the face using an algorithm based on the chosen method. If the object has not registered a face, the 'enroll' process must be done first to get the dataset, then perform the 'train' process to recognize faces according to machine learning algorithms. If the 'enroll' and 'train' processes have been carried out, the object can directly face its face to the camera and choose a facial recognition method to test the metric parameters that have been determined.

If a face (object) was identified in the dataset and detected using a mask, the Raspberry Pi will record the data in the database and send a command signal to the electric door driver (door lock) to open it. If the object is detected with a body temperature above 38^0C, the Raspberry Pi will immediately send a notification to the Android application installed on the cellphone of the relevant party so that COVID-19 prevention measures can be taken immediately.

B. Hardware Design

The hardware used in this research is:

- Rasbperry Pi 3B, Pi 4-4Gb, Pi 4-8Gb as CPU
- USB webcam 2 MP, to take pictures of face
- Relay, as a driver to move electric door lock
- Power supply 12V/5A, as main supply

The electric door lock works on the principle of an electromagnet, as depicted in the figure 2

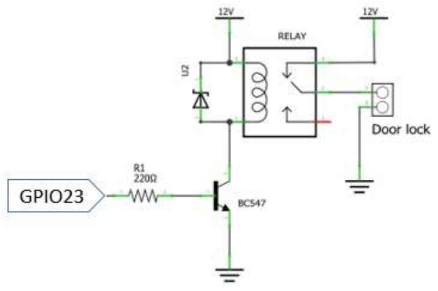

Figure 2. Electrik door lock schematic

If there is a voltage (DC 12V), the coil will function as a magnet that will pull the hook of the door.

C. Software Design

On the Raspberry Pi, the software used is:

- Linux distro Raspbian, as Operating System
- Python 3.8.1, as SDK (Software Development Kit)
- OpenCV 4.0, as Computer Vision library
- Haar cascade and MTCNN as face detection algorithm
- LBPH andTensorflow as face identification algorithm.

The flow chart for the software running on Raspberry can be shown in figure 3 below:

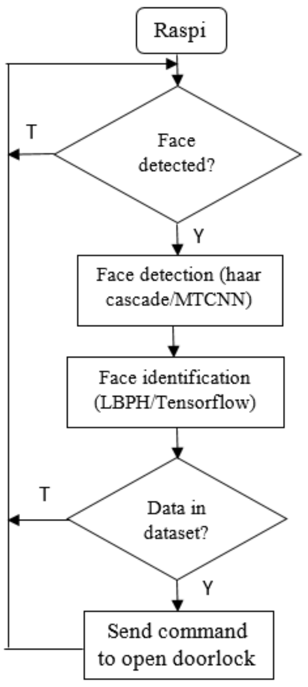
Raspi
Face detected?
T
Y
Face detection (haar cascade/MTCNN)
Face identification (LBPH/Tensorflow)
Data in dataset?
T
Y
Send command to open doorlock

Figure 3. Raspberry Pi flowchart for attendance system based facial recognition algorithm

D. Implementation System

The tools that have been made are then installed near the door of the 'MST' Lab in the Politeknik Negeri Semarang Masters building on the 2nd floor as the object of research. Tools are then tested based on predetermined testing metrics.

E. Testing System

The system is then tested by taking 10 respondents as objects (faces). Each respondent first carried out the learning process (train) data. After that, each object will be tested for its face recognition process one by one and its accuracy data will be taken. For other parameters (fps, CPU temperature) data is also recorded according to the data test metrics in table I.

TABLE I. DATA METRIC TESTING

CPU	Method	FPS	Accuration (%)	CPU Temp
Raspberry Pi 3B (core 1.2 GHz)	Haar Cascade+LBPH	From camera	From algorithm	From script
	Haar Cascade+Tensorflow	From camera	From algorithm	From script
	MTCNN+Tensorflow	From camera	From algorithm	From script
Raspberry Pi 4-4Gb (core 1.5 GHz)	Haar Cascade+LBPH	From camera	From algorithm	From script
	Haar Cascade+Tensorflow	From camera	From algorithm	From script
	MTCNN+Tensorflow	From camera	From algorithm	From script
Raspberry Pi 4-8Gb (core 1.5 GHz)	Haar Cascade+LBPH	From camera	From algorithm	From script
	Haar Cascade+Tensorflow	From camera	From algorithm	From script
	MTCNN+Tensorflow	From camera	From algorithm	From script

F. Maintenance System

System maintenance is conducted by continuing to test the system that has been made for a certain period of time so that the system's weaknesses can be identified.

III. Result And Discussion

A. Haar Cascade and LBPH Method

Face detection with Haar Cascade algorithm in OpenCV is done by calling the function `cv2.CascadeClassifier ("haarcascade_frontalface_default.xml"). detectMultiScale()`. The function call will produce the output in the form of the coordinates of the top left corner of the face (x,y) and the values of w (width) and h (height) as shown in Figure 4.

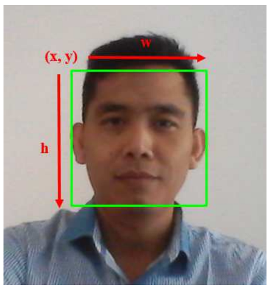

Figure 4. Face detection results with the Haar Cascade algorithm

After the face is detected, then the face is cut according to the parameters x, y, w and h, and then introduced (recognized) to the Raspberry Pi using the LBPH algorithm. In OpenCV, face recognition with LBPH can be done by calling the function `confidence = recognizer.predict (gray[y:y+h,x:x+w])`, with the output in the form of confidence value. After being recognized, the face will be searched for data in the dataset and its identity written on the monitor screen as shown in the picture 5

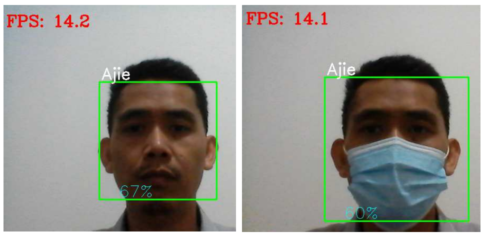

Figure 5. Results of face identification with Haar Cascade and LBPH . algorithms

B. Haar Cascade and Tensorflow Method

For the face detection method, the process is the same as the first method, but the identification process continues using the deep learning library Tensorflow.

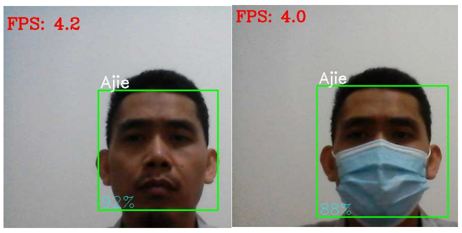

Figure 6. Results of face identification with Haar Cascade and Tensorflow algorithms

C. MTCNN and Tensorflow Method

Unlike the two previous methods, this third method uses a deep learning algorithm for both face detection and face identification processes.

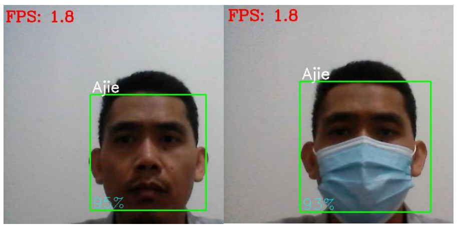

Figure 7. The results of face identification with the MTCNN and Tensorflow algorithms

D. Testing Result

From the results of testing 3 facial recognition methods running on 3 types of Raspberry Pi and 10 respondents, the results are shown in table II (the numbers listed in the table show the average value of all respondents tested). The data is then scored based on the level of significance to the system being tested to get the best method results on the most optimal Raspberry Pi hardware.

TABLE II. TESTING RESULT

CPU	Method	FPS	Accuration (%)	CPU Temp	Score
Raspberry Pi 3B (core 1.2 GHz)	Haar Cascade + LBPH	10.8	67	58^{0}C	94
	Haar Cascade + Tensorflow	4.0	92	64^{0}C	95
	MTCNN + Tensorflow	0.8	94	69^{0}C	90
Raspberry Pi 4-4Gb (core 1.5 GHz)	Haar Cascade + LBPH	14.2	67	58^{0}C	96
	Haar Cascade + Tensorflow	4.2	92	62^{0}C	94
	MTCNN + Tensorflow	1.8	95	66^{0}C	93
Raspberry Pi 4-8Gb (core 1.5 GHz)	Haar Cascade + LBPH	16	68	58^{0}C	93
	Haar Cascade + Tensorflow	4.5	92	62^{0}C	95
	MTCNN + Tensorflow	2.0	95	65^{0}C	94

From table II above, it is found that of the three methods applied to each type, the Raspberry Pi 4 with 8Gb RAM has the best performance among the CPU parameters tested (FPS and CPU temperature), while for the accuracy level parameters relative same. The next best performance in a row is the Raspberry Pi 4B with 4Gb RAM and the Raspberry 3B.

Meanwhile, in terms of methods, the Haar Casacade and LBPH algorithms are ranked first for testing the performance of artificial intelligence computational efficiency which is characterized by the best processing frames per second (FPS), which ranges from 10-14 fps for each type of Raspberry Pi. The next row is the Haar Cascade-Tensorflow method and the last one is MTCNN-Tensorflow. While testing the level of accuracy (confidency) shows the MTCNN-Tensorflow method has the best performance with an accuration rate above 95%.

IV. CONCLUSION

From the test data and scoring results provided, it is found that the combination of the Haar Cascade method and LBPH (Local Binary Pattern Histogram) is the most optimal method applied to the Raspberry Pi 4 with 4GB RAM as the hardware base used for attendance systems based on face recognition and masks.

For further research, the system needs to improve its accuracy level to above 90% and also improve its security side from trying to cheat the system with fake face images or videos, without increasing the computational burden on the Raspberry Pi hardware side.

BIBLIOGRAPHY

[1] D. Bhattacharyya, R. Ranjan, F. A. a, dan M. Choi, "Biometric Authentication : A Review," *Int. J. Serv. Sci. Technol.*, vol. 2, no. 3, hal. 13–28, 2009.

[2] M. Alhothaily, M. Alradaey, M. Oqbah, dan A. El-Kustaban, "Fingerprint Attendance System for Educational Institutes," *J. Sci. Technol.*, vol. 20, no. 1, hal. 34–44, 2015, doi: 10.20428/jst.20.1.4.

[3] O. O. Mikail dan B. U. Umar, "Design and Development of a Fingerprint Door," no. May, 2018.

[4] Https://www.solution.co.id/, "Fingerprint Time Attendance - Face Identification - RFID - Mifare." [Daring]. Tersedia pada: https://www.solution.co.id/en/c1.php. [Diakses: 28-Jan-2020].

[5] W. Yang, S. Wang, J. Hu, G. Zheng, dan C. Valli, "Security and accuracy of fingerprint-based biometrics: A review," *Symmetry (Basel).*, vol. 11, no. 2, 2019, doi: 10.3390/sym11020141.

[6] R. Yustiawati *et al.*, "Analyzing of Different Features Using Haar Cascade Classifier," *Proc. 2018 Int. Conf. Electr. Eng. Comput. Sci. ICECOS 2018*, vol. 17, hal. 129–134, 2019, doi: 10.1109/ICECOS.2018.8605266.

[7] D. E. King, "Dlib-ml: A machine learning toolkit," *J. Mach. Learn. Res.*, vol. 10, hal. 1755–1758, 2009.

[8] K. Zhang, Z. Zhang, Z. Li, dan Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," *IEEE Signal Process. Lett.*, vol. 23, no. 10, hal. 1499–1503, 2016, doi: 10.1109/LSP.2016.2603342.

[9] K. Jain, S. Narang, M. Saxena, dan A. Arora, "Comparison of Face Recognition Algorithms Using Opencv for Attendance System," *Int. J. Sci. Res. Publ.*, vol. 8, no. 2, hal. 268–273, 2018.

[10] I. Udlhiya, B. Supriyo, E. D. Wardihani, dan R. M. Firdaus, "Angular Detection System on Goal Frame using Image Processing with LabVIEW," *Proc. - 2019 Int. Semin. Appl. Technol. Inf. Commun. Ind. 4.0 Retrosp. Prospect. Challenges, iSemantic 2019*, hal. 277–281, 2019, doi: 10.1109/ISEMANTIC.2019.8884335.

[11] L. Dinalankara, "Face Detection & Face Recognition Using Open Computer Vision Classifies USING OPEN COMPUTER VISION," 2017.

[12] S. Emami dan V. P. Suciu, "Facial Recognition using OpenCV," *J. Mobile, Embed. Distrib. Syst.*, vol. 4, no. 1, hal. 38–43, 2012.

[13] A. Nag, J. N. Nikhilendra, dan M. Kalmath, "IOT Based Door Access Control Using Face Recognition," *2018 3rd Int. Conf. Converg. Technol. I2CT 2018*, hal. 1–3, 2018, doi:

10.1109/I2CT.2018.8529749.

[14] Y. Januzaj, A. Luma, Y. Januzaj, dan V. Ramaj, “Real Time Access Control Based on Face Recognition,” no. November, 2015, doi: 10.15242/iae.iae0615004.

[15] M. A. Akhloufi dan A.-C. Guei, “Deep learning for face recognition at a distance,” no. July, hal. 25, 2018, doi: 10.1117/12.2304896.

[16] S. Liao, A. K. Jain, dan S. Z. Li, “A Fast and Accurate Unconstrained Face Detector,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, hal. 211–223, 2016, doi: 10.1109/TPAMI.2015.2448075.