Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)
Vol. 8, No. 3, September 2022, pp. 399-409
ISSN: 2338-3070, DOI: 10.26555/jiteki.v8i3.24457

Optimize Image Processing Algorithm on ARM Cortex-A72 and A53

Rachmat Muwardi [1], Mirna Yunita [2], Harun Usman Ghifarsyam [3], Hendy Juliyanto [4]
[1] Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana, Indonesia
[2] School of Computer Science and Technology, Beijing Institute of Technology, China
[3] School of Computer and Information Technology, Beijing Jiaotong University, China
[4] Department of Information System, Faculty of Computer Science, Universitas Mercu Buana

ARTICLE INFO

Article history:

Received July 16, 2022
Revised August 23, 2022
Published September 14, 2022

Keywords:

Multi-core Processing;
Real-time System;
Computer Vision;
Image Processing;
Image Feature Extraction;
NanoPi M4V2;
Embedded System

ABSTRACT

This work presents a technique to optimize processing image algorithms. The increasing demand for video applications like context-aware computing on mobile embedded systems requires the use of computationally intensive image processing algorithms. The system engineer has the mandate to optimize them so as to meet real-time deadlines. A methodology to take advantage of the asymmetric dual-core processor, which includes an ARM supported by shared memory, is presented with implementation details. The target platform chosen is the NanoPi M4V2. It has a dual-core and quad-core architecture with an ARM Cortex-A72 and Cortex-A53. The basic image correlation algorithm is chosen for benchmarking as it finds widespread application for various template-matching tasks such as face recognition. The basic algorithm prototypes conform to OpenCV, a popular computer vision library. OpenCV algorithms can be easily ported to the ARM core, which runs a popular operating system such as Linux. The algorithms are tested on a variety of images, and performance results are presented, measuring the speedup obtained due to dual-core and quad-core implementation. A major advantage of this approach is that it allows the ARM processor to perform important real-time tasks.

Corresponding Author:

Rachmat Muwardi, Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana, Indonesia
Email: rachmat.muwardi@mercubuana.ac.id

1. INTRODUCTION

In recent years, the study of image analysis and its use in face recognition applications has gained significant attention from the worldwide research community. Face recognition has been widely used as identification, and it has become a special application among many fields of research in Computer Vision and pattern recognition, such as commercial and law enforcement applications, including passports, credit cards, driver's licenses, biometric authentication, video surveillance, and information security.

Face Recognition is a recognition technique used to detect the faces of individuals whose images are saved in the data set. Before it can be identified, first, the face in the image should be located. Unlike the human eye and brain, a computer cannot see a visual object as a human does. The computer will see the object as binaries strings [1-5].

A facial recognition system is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from a given image with faces within a database. It is also described as a Biometric Artificial Intelligence based application that can uniquely identify a person by analyzing patterns based on the person's facial textures and shape [6].

While initially a form of computer application, it has seen wider uses in recent times on mobile platforms and in other forms of technology, such as robotics. It is typically used as access control in security systems and can be compared to other biometrics, such as fingerprint or eye iris recognition systems. Although the accuracy of facial recognition systems as a biometric technology is lower than iris recognition and fingerprint recognition, it is widely adopted due to its contactless and non-invasive process [7-9]. Recently, it has also become popular as a commercial identification and marketing tool. Other applications include advanced human-computer interaction, video surveillance, automatic indexing of images, and video database, among others [10-13].

RK3399 is a low-power, high-performance processor for computing, personal mobile internet devices, and other smart device applications. Based on Big. Little architecture, it integrates dual-core Cortex-A72 and quad-core Cortex-A53 with a separate NEON coprocessor. It also integrates Mali T860 MP4 GPU.

Many embedded powerful hardware engines provide optimized performance for the high-end application. RK3399 supports multi-format video decoders, including H.264/H.265/VP9 up to4Kx2K@60fps, especially H.264/H.265 decoders support 10bits coding, and also supports H.264/MVC/VP8 encoders by 1080p@30fps, high-quality JPEG encoder/decoder, and special image preprocessor and postprocessor.

Embedded 3D GPU makes RK3399 completely compatible with OpenGL ES1.1/2.0/3.0/3.1, OpenCL, and DirectX 11.1. A special 2D hardware engine with MMU will maximize display performance and provide very smooth operation.

RK3399 has a high-performance dual-channel external memory interface (DDR3/ DDR3L/ LPDDR3/ LPDDR4) capable of sustaining demanding memory bandwidths and also provides a complete set of the peripheral interface to support very flexible applications.

The embedded face recognition system takes ARM as the core control unit to realize facial feature extraction and image recognition. The embedded face recognition system designed uses a modular platform design, mainly divided into a multi-modal state monitoring sensor module and communication module, power supply module, PC communication module, the data processing module, and DSP digital signal processing module, which DSP (digital signal processor) is the core of the system, ARM is used as the main control chip of the control system, WiFi is used for data transmission and control command output, liquid crystal display is used to achieve human-computer interaction [14-16].

2. METHODS

After considering all the background discussed, we can conclude to solve this problem by using Multi-thread. Then there will be some problems and questions about this multi-thread, such as what image want to process, what is the purpose of image processing, How to process images, how to make a detailed and correct image processing framework design, make the core work regularly and directed, and utilize ARM tools with high efficiency when using the image processing framework real-time. All the problems mentioned above will be resolved and will be explained in the next section [17-19].

This work presents a methodology for the optimization of image processing on ARM. Every Biometric system has four main features, which are shown in Fig. 1, face Detection, preprocessing, Feature Extraction, and Face Recognition.

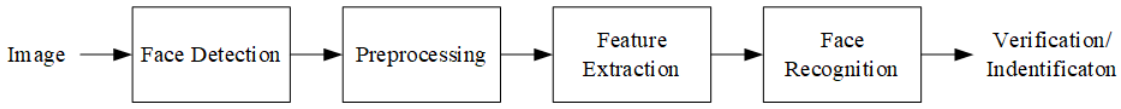

Fig. 1. Architecture of Face Recognition System

As Fig. 1. shows, the first task of the face recognition system is capturing an image by video, camera, or from the database, and this image is given to the further step of the face recognition system that is discussed in this section [20-22].

Face Detection-The main function of this step is to detect the face from the captured image or the selected image from the database. This face detection process actually verifies whether the given image has a face image or not. After detecting the face, this output will be further given to the pre-processing step.

The pre-processing-This step works as the pre-processing for face recognition. In this step, the unwanted noise, blur, varying lighting conditions, and shadowing effects can be removed using pre-processing techniques. Once we have a fine smooth face image, then it will be used for the feature extraction process.

Feature Extraction-In this step, features of the face can be extracted using a feature extraction algorithm. Extractions are performed to do information packing, dimension reduction, salience extraction, and noise-

cleaning. After this step, a face patch is usually transformed into a vector with a fixed dimension or a set of fiducial points and their corresponding locations.

Face Recognition-Once feature extraction is done step analyzes the representation of each face; this last step is used to recognize the identities of the faces for achieving automatic face recognition. For recognition, a face database is required to build. For each person, several images are taken, and their features are extracted and stored in the database. Then when an input face image comes for recognition, then it first performs face detection, preprocessing, and feature extraction. After that, it compares its feature to each face class stored in the database.

Digital image processing is the use of a digital computer to process digital images through an algorithm [23][24]. As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and distortion during processing since images are defined over two dimensions (perhaps more). Digital image processing may be modeled in the form of multidimensional systems. The generation and development of digital image processing are mainly affected by three factors: first, the development of computers; second, the development of mathematics (especially the creation and improvement of discrete mathematics theory); third, the demand for a wide range of applications in environment, agriculture, military, industry, and medical science has increased.

Real-time computing (RTC), or reactive computing, is the computer science term for hardware and software systems subject to a "real-time constraint," for example, from event to system response. Real-time programs must guarantee a response within specified time constraints, often referred to as "deadlines."

Real-time responses are often understood to be in the order of milliseconds and sometimes microseconds. A system not specified as operating in real-time cannot usually guarantee a response within any timeframe, although typical or expected response times may be given. Real-time processing fails if not completed within a specified deadline relative to an event; deadlines must always be met, regardless of system load.

Advanced RISC Machine (ARM) Processor is a family of reduced instruction set computing (RISC) architectures for computer processors configured for various environments. Arm Holdings develops the architecture and licenses it to other companies, who design their own products that implement one of those architectures—including systems-on-chips (SoC) and systems-on-modules (SoM) that incorporate memory, interfaces, radios, etc. It also designs cores that implement this instruction set and licenses these designs to a number of companies that incorporate those core designs into their own products.

OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision [25-34]. Originally developed by Intel, it was later supported by Willow Garage, then Itseez.

Data acquisition techniques can be divided into two categories: Active Range Sensors and Passive Range Sensors, that is, data acquisition based on physical devices and data acquisition based on multiple images [35-40]. Active Ranging Sensing is when the vision system first emits energy to the scene and then receives the energy emitted by the scene to calculate the location information of each point in the scene. A passive ranging sensor is a visual system that receives the light energy emitted or reflected from the scene, forms the light energy distribution function of the scene, and then restores the depth information of the scene on the basis of these images. The most common method is to use two cameras at a certain distance to obtain scene images simultaneously to generate depth maps. Another method similar to this method is that a camera obtains two or more images in different spatial positions and generates depth maps by gray information and imaging geometry of multiple images. Depth information can also be indirectly estimated by using the gray and dark features, texture features, and motion features of grayscale images. The methods used in the completion of the research are written in this section.

As it was proven that Nanopi M4V2 is capable of handling this experiment, so there is huge potential for this research. The code can be improved by a better algorithm and better detection parameters. The captured video is also possible to store for later processing. The device can handle more complex and demanding tasks. So it is also possible to decentralize the image processing task into individual devices. The device is capable of being assigned for more instruction and external commands regarding the respective use case.

2.1. Workflow Design

There is a framework part, as it is required for building and enabling face recognition optimization algorithm (Fig. 2). The other is analysis, which shows the difference between unoptimized and optimized ones. Feature location of the face image is taken by image edge contour detection method, combined with image entropy adaptive feature extraction method, face feature extraction is realized, and the face recognition algorithm is improved. In order to achieve effective extraction and recognition of facial feature points, an

adaptive point tracking method is used to label and detect the region of facial activity, and the face feature points are constructed as the coordinate of the facial feature points:

$$X = (x_{i0}, x_{i1}, \dots, x_{i(n-1)}, y_{i0}, y_{i1}, \dots., y_{i(n-1)})^T \quad (1)$$

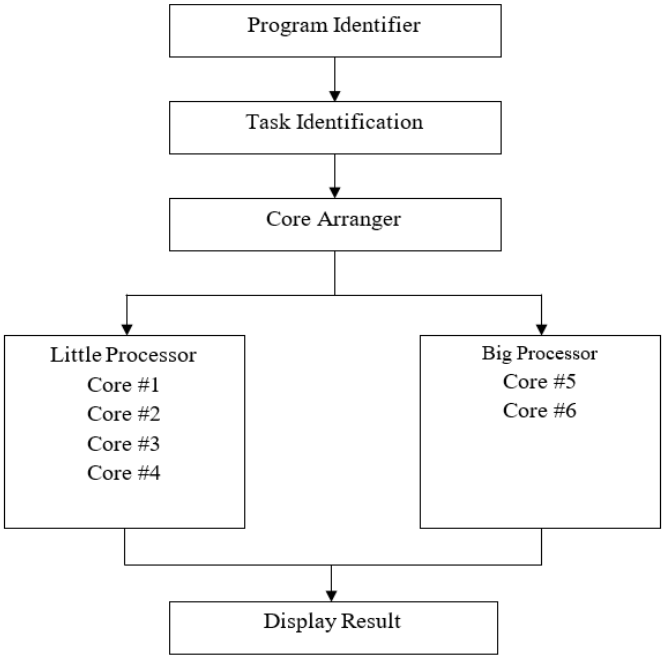

Fig. 2. Parallelism Scheme

The affine transformation of the facial feature points is used to analyze the affine transformation, and the affine transformation coordinate system is constructed as $X = (x_{i0}, x_{i1}, \dots, x_{i(n-1)}, y_{i0}, y_{i1}, \dots., y_{i(n-1)})^T$. By tracking facial feature points to calculate the j gradient histogram of all samples in face images, in the pixels (x_{ij}, y_{ij}), by an affine transformation, get the corresponding invariant coordinates (x_{ij}, y_{ij}), face mesh region is matched in different orientations and different scales, the facial details are highlighted, and the relative gradient values of face images are:

$$err_{ij} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_{ij} - x_{ij})^2 + (y_{ij} - y_{ij})^2} \quad (2)$$

Where N is the total number of pixels of the face image, and ij is the number of corresponding facial features, the feature matrix of face pose change during rotation is:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(-\theta_1) & -\sin(-\theta_1) & 0 \\ \sin(-\theta_1) & \cos(-\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (3)$$

The affine invariant moment matrix is obtained by adjusting the pixel value in the original image, and the affine invariant matrix of the face pose change contour is obtained in the projection coordinate:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (4)$$

In the formula, EyeMapCx is the image information entropy feature, the affinity matrix of facial features is calculated, and thereby the feature extraction and face recognition are completed.

Initial setting handler

This function is necessary for creating the initial setting or applying the saved setting. This function will also handle various system calls needed for the initial start of the framework. Optionally, this will also contain the necessary function for analysis purposes as a time recorder.

Processor core and hardware identification

The framework needs to identify the specification of the processor. Then this function will create the configuration file of the hardware that will be utilized by the framework. The function will also prepare the necessary system call and hardware utilization function for the other entity to work on.

Image retrieval by the camera

Image data will be retrieved by this function inside the framework. This function will analyze and settle the hardware communication and system call that is necessary for the framework.

Core utilization analysis

This function will actively monitor the processor's utilization data and determine which core is available for the task. We can set the policy for each core's utilization. For example, the core with high utilization will not be utilized for the task. This function also provides data for the workload assigner function.

Workload division mapping and assigner for each core

This function will receive data from the core utilization analysis function. This will determine which task will be executed by which core. This function will manage the assignment on the OS layer and process utilization, so it can pinpoint to which core the given task will run.

Execute image processing task

The image processing function will run at the given core.

Retrieve and arrange the raw result and data of the task

Because of the parallelization, the result of each core's process will be retrieved, and those will be combined into necessary data for further processing

Further result processing

This function will handle the result finalization and data shaping as intended.

Result analysis

The finalized data and running time of one cycle of the framework will be checked by this function. This function will also handle functions necessary for analysis purposes.

2.2. Hardware Design

The schematics of the device are shown in Fig. 3. The device is connected to the network, USB Camera (C270), and SSD through a GPIO pin. Then for the native display, it connects to the LG monitor through an HDMI port.

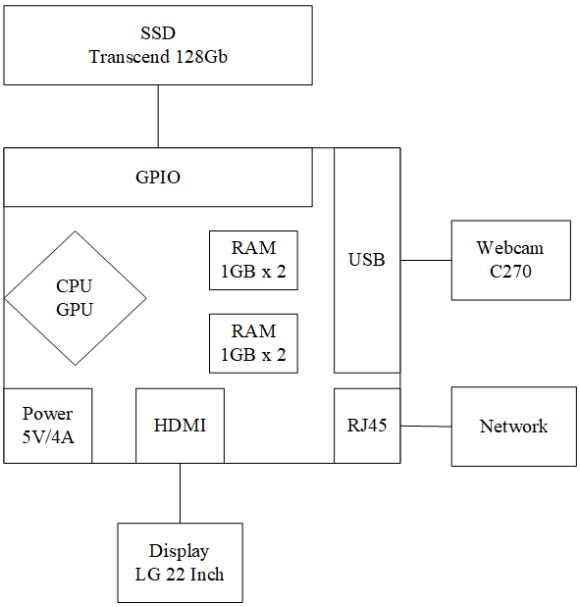

Fig. 3. Hardware Design

The program design is shown in Fig. 4. Firstly, the program needs to import the library, then load the necessary dataset. In python, we can identify whether the code is valid or not by just running the code by itself. Then we put the loop "While true" to avoid exiting by itself. Inside the loop, we put the core functionality. It captures video by frame, detects multi-scale, draws a rectangle around the face, and displays the resulting frame.

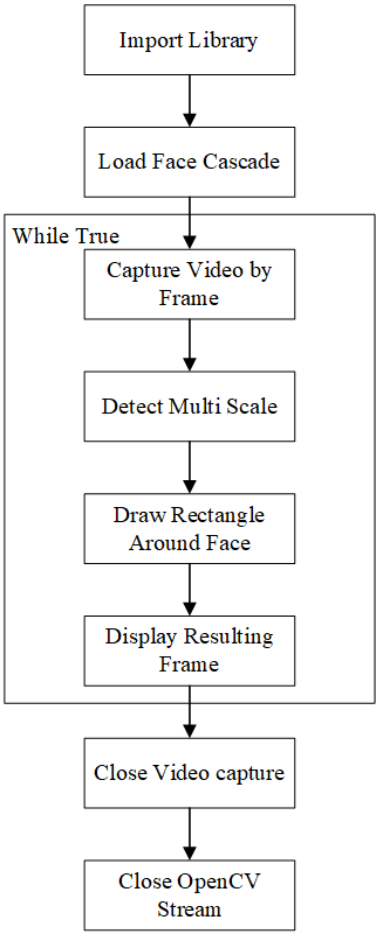

Fig. 4. Program Architecture

Then we can put the termination code inside the loop and wrap the program up at the end of the code. Fig. 5 shows the flow of the data, starting from a USB camera that captures the video and passes it into the device. Then, the device will process the data and finally shows the result on display.

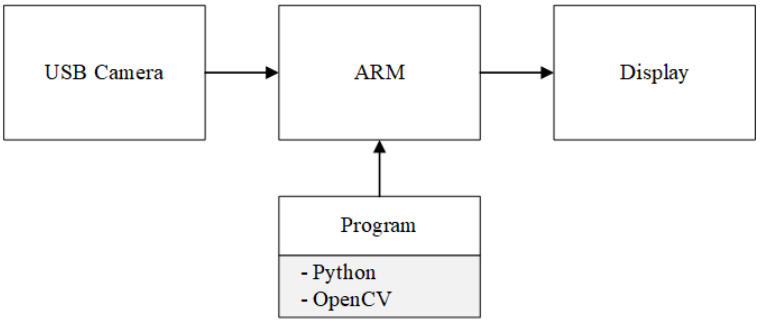

Fig. 5. Processing

Face detection is the most part of face recognition which detects faces from video or image. In [41], it is given popular face detection algorithms. Classification of face detection algorithms is face detection by skin color, face detection by Haar classifier, and face detection by face characteristics.

Table 1 shows problems can be due to system faults used in face recognition, such as camera distortion, background noise, inefficient storage, improper techniques, etc. More than that, there can be network problems due to environmental conditions.

Table 1. Analyzes face detection method by detection rate and time

Face Detection Method	Detection Rate	Detection Time
Skin Color	66%	0.410 s
Haar Classifer	87%	0.443 s
Face Characteristics	93%	1.669 s

3. RESULTS AND DISCUSSION

Then the program will capture the video and divide it by frame. So, the OpenCV library can work with image format. The frame in the current process will be transformed into grey. We need to perform the detectMultiScale function. There are several parameters needed for that function. It is the input file, scaleFactor, minNeighbors, minSize, and flags. The designated face detection program runs smoothly in Nanopi M4V2 are shown in Fig. 6.

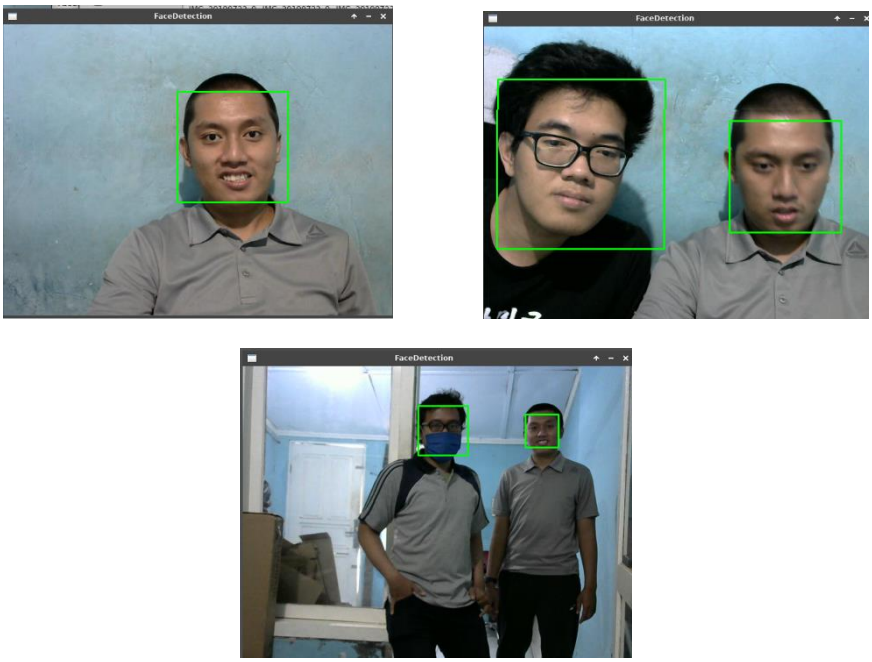

Fig. 6. Result in Detection

There are no errors in the process because the program does not output any error message or throw any interruptions during the test. There is also no stuttering or undesired behavior, shown in the resource manager that the CPU utilization is still below 80 % while running the program. The resources needed for the program to run is also adequate in the given device, as the requirement to run the program is fulfilled by the hardware specification of the device.

The detection is relatively inaccurate due to unrefined code. It is possible to improve the accuracy of the code without sacrificing performance. The test also showed that OpenCV and python three runs without problem in Fig. 7 and Fig. 8, as no interruptions or errors were present during the testing. There is no detected bottleneck in the performance. Native display and remote access worked as intended in the program environment for the hardware itself. The experiment does not cause overheating even without an active device cooling mechanism. No hardware failures or slowdowns happened due to the performance of the given specification. Peripherals such as USB Cameras, SSD, and displays worked without a problem.

They utilize image processing algorithms and ARM devices with high efficiency when using face detection, recognition, and image processing framework in real time. For hardware utilization, RAM usage is relatively low, just around 200 MB with an occasional spike up to 300 MB. Processor utilization for this program takes about 50 % of the processor's capability. The final results of this experiment occasionally spike to 100 % CPU usage. Table 2 shows the performance analysis of The Nanopi M4V2.

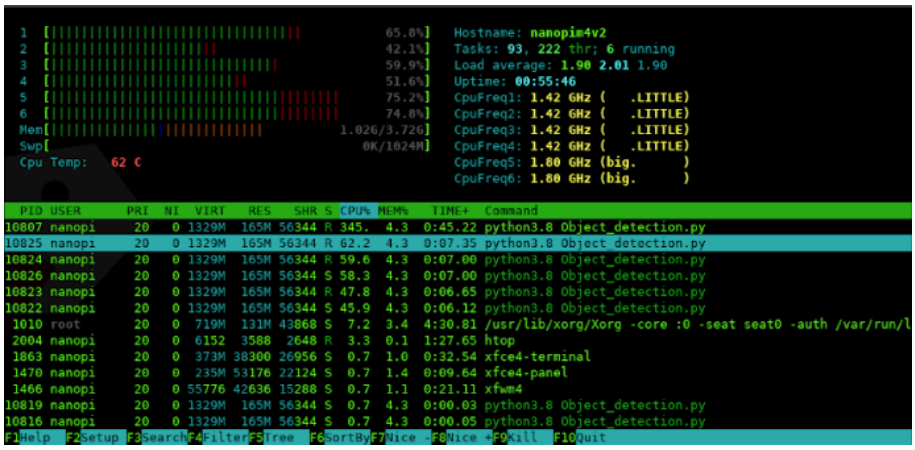

Fig. 7. While running the program.

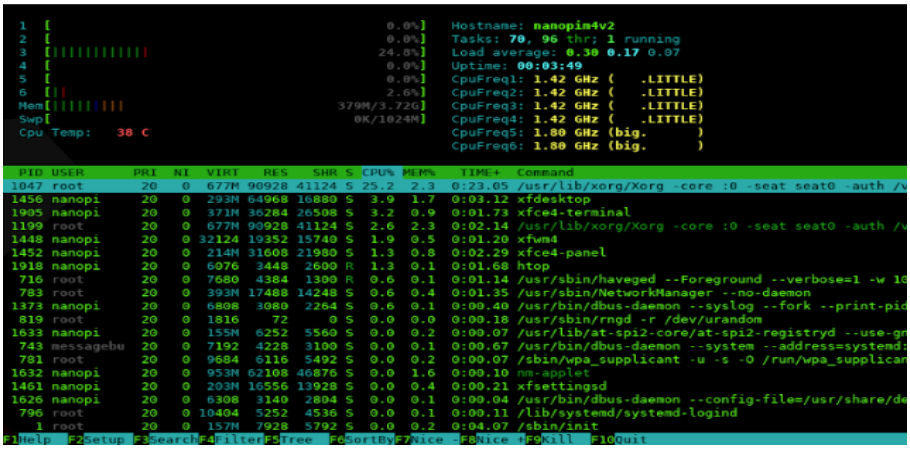

Fig. 8. Without running the program.

Table 2. Performance Analysis

Name	CPU Frequency	Without Running Program	While Running Program
	1.42 GHz	0%	65.8%
Little	1.42 GHz	0%	42.1%
Core	1.42 GHz	24.0%	59.9%
	1.42 GHz	0%	51.6%
Big	1.8 GHz	0%	75.2%
Core	1.8 GHz	2.6%	74.8%
RAM	4 GB	10%	75%
CPU Temp		38C°	62C°

4. CONCLUSION

The researcher wrote hopes to design an image processing framework in detail and correctly, make the core work regularly and directed, and utilize ARM devices with high efficiency when using an image processing framework in real-time. This system is organized to capture an image sequence, find facial features in the images, and recognize and verify a person. In order to improve the ability of face recognition and detection, an improved face recognition system is proposed based on ARM architecture design, and software development of face recognition system is taken in ARM embedded platform. The image edge contour detection method is used for face image feature positioning, the adaptive feature extraction method of image entropy is used for facial feature extraction, the face recognition algorithm is optimized, and the user application program, the bus driver module, the information interaction module, and the face image output module are constructed. The application development of a face recognition system is taken based on ARM architecture, and it can improve the accurate probability of face recognition. In general, facial recognition systems proceed by capturing the face in an image, with the effect of estimating and normalizing for translation, scale, and in-plane rotation. Given a normalized image, the features are extracted and compressed in a compact

face representation which can then be stored in an image database and compared with face representations image derived at later times.

Acknowledgments

The first special thanks go to Universitas Mercu Buana for supporting foreign collaborative research, and the second to Beijing Institute of Technology Mirna Yunita for their help and cooperation during this research. Third to Beijing Jiaotong University, Harun Usman Ghifarsyam, for providing unfailing support and continuous encouragement. Hopefully, the collaboration in publishing papers will continue with the Beijing Institute of Technology and Beijing Jiaotong University in further research.

REFERENCES

[1] R. Muwardi, H. Qin, H. Gao, H. U. Ghifarsyam, M. H. I. Hajar and M. Yunita, "Research and Design of Fast Special Human Face Recognition System," *n 2020 2nd International Conference on Broadband Communications, Wireless Sensors and Powering (BCWSP)*, pp. 68-73, 2020, https://doi.org/10.1109/BCWSP50066.2020.9249452.

[2] D. N. Parmar and B. B. Mehta, "Face Recognition Methods & Applications," *arXiv preprint arXiv:1403.0485*, 2014, https://doi.org/10.48550/arXiv.1403.0485.

[3] M. Khosravy, K. Nakamura, Y. Hirose, N. Nitta, and N. Babaguchi, "Model Inversion Attack by Integration of Deep Generative Models: Privacy-Sensitive Face Generation from a Face Recognition System," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 357-372, 2022, https://doi.org/10.1109/TIFS.2022.3140687.

[4] M. Tamilselvi and S.Karthikeyan, "An ingenious face recognition system based on HRPSM_CNN under unrestrained environmental condition," *Alexandria Engineering Journal*, vol. 61, no. 6, pp. 4307-4321, 2022, https://doi.org/10.1016/j.aej.2021.09.043.

[5] M. N. Favorskaya and A. I. Pakhirka, "Image-based anomaly detection using CNN cues generalisation in face recognition system," *International Journal of Reasoning-based Intelligent Systems*, vol. 14, no. 1, pp. 19-26, 2022, https://doi.org/10.1504/IJRIS.2022.123389.

[6] S. Albalawi, L. Alshahrani, N. Albalawi, R. Kilabi, and A. Alhakamy, "A Comprehensive Overview on Biometric Authentication Systems using Artificial Intelligence Techniques," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 4, 2022, https://doi.org/10.14569/IJACSA.2022.0130491.

[7] W. M. Hammond, A. P. Williams, J. T. Abatzoglou, H. D. Adams, T. Klein, R. López, C. Sáenz-Romero, H. Hartmann, D. D. Breshears, and C. D. Allen, "Global field observations of tree die-off reveal hotter-drought fingerprint for Earth's forests," *Nature Communications*, vol. 13, no. 1, pp. 1-11, 2022, https://doi.org/10.1038/s41467-022-29289-2.

[8] T. Jucker, "Deciphering the fingerprint of disturbance on the three-dimensional structure of the world's forests," *New Phytologist*, vol. 233, no. 2, pp. 612-617, 2022, https://doi.org/10.1111/nph.17729.

[9] F. Saeed, M. Hussain, and H. A. Aboalsamh, "Automatic Fingerprint Classification Using Deep Learning Technology (DeepFKTNet)," *Mathematics*, vol. 10, no. 8, p. 1285, 2022, https://doi.org/10.3390/math10081285.

[10] R. Ding, H.-b. Dong, G.-s. Yin, J. Sun, X.-d. Yu and X.-b. Feng, "An objective reduction method based on advanced clustering for many-objective optimization problems and its human-computer interaction visualization of pareto front," *Engineering*, vol. 93, p. 107266, 2021, https://doi.org/10.1016/j.compeleceng.2021.107266.

[11] R. Muwardi, H. Gao, H. U. Ghifarsyam, M. Yunita, A. Arrizki, and J. Andika, "Network Security Monitoring System Via Notification Alert," *Journal of Integrated and Advanced Engineering (JIAE)*, vol. 1, no. 2, 2021, https://doi.org/10.51662/jiae.v1i2.22.

[12] R. Xu and H. Chen, "Application of Human-Computer Interaction Based on Big Data Technology in Electronic Product Design," *Journal of Physics: Conference Series*, vol. 1992, no. 2, p. 022014, 2021, https://doi.org/10.1088/1742-6596/1992/2/022014.

[13] M. Pikhart, "Human-computer interaction in foreign language learning applications: Applied linguistics viewpoint of mobile learning," *Procedia Computer Science*, vol. 184, pp. 92-98, 2021, https://doi.org/10.1016/j.procs.2021.03.123.

[14] X. Lv, M. Su, and Z. Wang, "Application of Face Recognition Method Under Deep Learning Algorithm in Embedded Systems," *Microprocessors and Microsystems*, vol. 104034, 2021, https://doi.org/10.1016/j.micpro.2021.104034.

[15] J. Wan, Y. Chen and B. Bai, "Joint feature extraction and classification in a unified framework for cost-sensitive face recognition," *Pattern Recognition*, vol. 115, p. 107927, 2021, https://doi.org/10.1016/j.patcog.2021.107927.

[16] J. Gao, M. Xu, H. Wang, and J. Zhou, "End-to-end Saliency Face Detection and Recognition," *Journal of Physics: Conference Series*, vol. 2171, no. 1, p. 012004, 2022, https://doi.org/10.1088/1742-6596/2171/1/012004.

[17] X. Chen, "Framework design of sports image analysis system based on three-dimensional image technology," *Journal of Physics: Conference Series*, vol. 1982, no. 1, p. 012208, 2021, https://doi.org/10.1088/1742-6596/1982/1/012208.

[18] G. Sharov, D. R. Morado, M. Carroni and J. M. d. l. Rosa-Trevín, "Using RELION software within the Scipion framework," *Acta Crystallographica Section D: Structural Biology*, vol. 77, no. 4, pp. 403-410, 2021, https://doi.org/10.1107/S2059798321001856.

[19] I. J. Jacob and P. E. Darney, "Design of Deep Learning Algorithm for IoT Application by Image based Recognition," *Journal of ISMAC*, vol. 3, no. 03, pp. 276-290, 2021, https://doi.org/10.36548/jismac.2021.3.008.

[20] S. Kim, Y. Jeong, J. Kim, J. Kim, H. T. Lee, and J. H. Seo, "IronMask: Modular Architecture for Protecting Deep Face Template," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, https://doi.org/10.1109/CVPR46437.2021.01586.

[21] O. Matei, R. Erdei, A. Moga and R. Heb, "A Serverless Architecture for a Wearable Face Recognition Application," 2021, https://doi.org/10.1007/978-3-030-68787-8_46.

[22] D. Deb, S. Wiper, S. Gong, Y. Shi, C. Tymoszek, A. Fletcher and A. K. Jain, "Face Recognition: Primates in the Wild," in *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, 2018, https://doi.org/10.1109/BTAS.2018.8698538.

[23] M. Mamatov and X. Alimov, "Differential Games and Methods of Digital Image Processing," *Journal of Physics: Conference Series*, vol. 1646, no. 1, p. 012020, 2020, https://doi.org/10.1088/1742-6596/1646/1/012020.

[24] V. V. Bodryshev, A. V. Babaytsev, and L. N. Rabinskiy, "Investigation of processes of deformation of plastic materials with the help of digital image processing," *Periodico Tche Quimica*, vol. 16, no. 33, pp. 865-876, 2019, https://doi.org/10.52571/PTQ.v16.n33.2019.880_Periodico33_pgs_865_876.pdf.

[25] S. Parveen and J. Shah, "A Motion Detection System in Python and Opencv," in *2021 third international conference on intelligent communication technologies and virtual mobile networks (ICICV)*, 2021, https://doi.org/10.1109/ICICV50876.2021.9388404.

[26] W. Sriratana, S. Mukma, and N. Tammarugwattana, "Application of the OpenCV-Python for Personal Identifier Statement," in *2018 International Conference on Engineering, Applied Sciences, and Technology (ICEAST)*, Phuket, Thailand, 2018, https://doi.org/10.1109/ICEAST.2018.8434429.

[27] M. Khan, S. Chakraborty, R. Astya and S. Khepra, "Face Detection and Recognition Using OpenCV," in *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, Greater Noida, India, 2019, https://doi.org/10.1109/ICCCIS48478.2019.8974493.

[28] G. Chandan, R. V. C. o. E. B. I. Ayush Jain Telecommunication Engineering, H. Jain and Mohana, "Real Time Object Detection and Tracking Using Deep Learning and OpenCV," in *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, India, 2018, https://doi.org/10.1109/ICIRCA.2018.8597266.

[29] K. Goyal, K. Agarwal, and R. Kumar, "Face detection and tracking: Using OpenCV," in *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*, 2017, https://doi.org/10.1109/ICECA.2017.8203730.

[30] H. Adusumalli, D. Kalyani, R. Sri, M. Pratapteja and P. V. R. D. P. Rao, "Face Mask Detection Using OpenCV," in *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, 2021, https://doi.org/10.1109/ICICV50876.2021.9388375.

[31] N. Boyko, O. Basystiuk and N. Shakhovska, "Performance Evaluation and Comparison of Software for Face Recognition, Based on Dlib and Opencv Library," in *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*, 2018, https://doi.org/10.1109/DSMP.2018.8478556.

[32] R. R. Palekar, S. U. Parab, D. P. Parikh, and V. N. Kamble, "Real time license plate detection using openCV and tesseract," in *2017 International Conference on Communication and Signal Processing (ICCSP)*, 2017, https://doi.org/10.1109/ICCSP.2017.8286778.

[33] C. V. Krishna, H. R. Rohit, and Mohana, "A Review of Artificial Intelligence Methods for Data Science and Data Analytics: Applications and Research Challenges," in *2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2018 2nd International Conference on*, 2018, https://doi.org/10.1109/I-SMAC.2018.8653670.

[34] A. Biswas, A. P. Jana, Mohana, and S. S. Tejas, "Classification of Objects in Video Records using Neural Network Framework," in *2018 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2018, https://doi.org/10.1109/ICSSIT.2018.8748560.

[35] M. R. Nehashree, P. R. S and Mohana, "Simulation and Performance Analysis of Feature Extraction and Matching Algorithms for Image Processing Applications," in *2019 International Conference on Intelligent Sustainable Systems (ICISS)*, 2019, https://doi.org/10.1109/ISS1.2019.8907990.

[36] S. Hu, "Research on Data Acquisition Algorithms Based on Image Processing and Artificial Intelligence," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 34, no. 06, p. 2054016, 2020, https://doi.org/10.1142/S0218001420540166.

[37] C. L. E. Germán González, "Biomedical Image Processing with Containers and DeepLearning: An Automated Analysis PipelineData architecture, artificial intelligence, automated processing, containerization, and clusters

orchestration ease the transition from data acquisition to insights," *BioEssays*, vol. 41, no. 6, p. 1900004, 2019, https://doi.org/10.1002/bies.201900004.

[38] Y. He and S. Chen, "Recent Advances in 3D Data Acquisition and Processing by Time-of-Flight Camera," *IEEE Access*, vol. 7, pp. 12495-12510, 2019, https://doi.org/10.1109/ACCESS.2019.2891693.

[39] X. He, J. Shao and J. Zhu, "Research on Digital Image Recognition Algorithm Based on Modular Intelligent Image Recognition," in *2021 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, 2021, https://doi.org/10.1109/AEECA52519.2021.9574139.

[40] Y. Yu, "Application of Smart Image Processing Technology in Feature Extraction of Glass Artistic Style Patterns," in *2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2022, https://doi.org/10.1109/ICSSIT53264.2022.9716431.

[41] A. Kumar, A. Kaur, and M. Kumar, "Face detection techniques: a review," *Artificial Intelligence Review*, vol. 52, no. 2, pp. 927-948, 2019, https://doi.org/10.1007/s10462-018-9650-2.

BIOGRAPHY OF AUTHORS

Rachmat Muwardi is currently a Lecturer in the Department of Electrical Engineering, Universitas Mercu Buana, Jakarta, Indonesia. He graduated from the Beijing Institute of Technology in 2020 with a Master's degree in Electronic Science and Technology. Currently, declared as a recipient of a China Scholarship Council (CSC) to continue his doctoral program at the Beijing Institute of Technology in September 2022, majoring in Optical Engineering. During his undergraduate, he received a double degree scholarship from Universitas Mercu Buana and Beijing Institute of Technology in Electrical Engineering and Computer Science. His research interests are Object Detection, Target Detection, Embedded Systems, and Network Security. It has been the basic area of my research as long as I have been studying so far. E-Mail: rachmat.muwardi@mercubuana.ac.id

Mirna Yunita received a Master's degree in Computer Science and Technology from the Beijing Institute of Technology, Beijing, China. She is currently a Frontend and Mobile App Developer in a Logistics & Supply Chain company in Jakarta, Indonesia. Her areas of interest include Machine Learning, Web Development, Data Mining, and Bioinformatics. E-Mail: mirnayunitaa@gmail.com

Harun Usman Ghifarsyam received a bachelor's degree from the Beijing Institute of Technology in Computer Science and Technology. Currently, he is pursuing a master's degree at Beijing Jiaotong University in Computer Technology. His research interest includes a microprocessor, image processing, and artificially intelligent. He currently works at ONDELIVERY Indonesia as Head of IT. E-Mail: harunsky@bjtu.edu.cn

Hendy Juliyanto was born in Jakarta (Indonesia) on 24 July 1996. He currently studies Information Systems major at Universitas Mercu Buana and works at PT. ONDELIVERY Since 2019 as VC Head IT. E-Mail: juliyantohendy@gmail.com