HUMAN RECOGNITION WITH YOLO TO REDUCE FALSE ALARMS IN THE INTERNET-OF-THINGS-BASED MOTION DETECTION SYSTEM

Galih Setiarso[1], Alauddin Maulana Hirzan[1] and Agus Hartanto[1]

[1]Fakultas Teknologi Informasi dan Komunikasi,
Universitas Semarang, Tlogosari Kulon, 50196 Kota Semarang, Jawa Tengah, Indonesia.

Corresponding Author's Email: [1]maulanahirzan@usm.ac.id

Article History: Received 02 January 2023; Revised 28 February 2023; Accepted 03 April 2023

ABSTRACT: Security is a serious matter that must not be ignored. When dealing with a Smart Home or Office, an Internet of Things-based approach is preferable to safeguard the physical system. Smart security devices offer many features to help homeowners to secure their houses. However, there are several issues in terms of storage and motion detection since motion detection cannot recognize what or who caused the movements, and unencrypted data is vulnerable to theft. In fact, the increasing exposure of the detected motions produces false alarms and false reports. Therefore, this study implements human recognition by adding redundant Blockchain and YOLO algorithms in motion detection. The findings have shown in the evaluation process that the proposed model only needed 58.362% of the CPU and 110.123MB of memory allocation. Meanwhile, the proposed model temperature reached 32.33 degrees Celsius on average. As a result, the execution of the proposed model, which is the human recognition with the YOLO algorithm successfully lowered the false alarms over the previous model that is deployed without the YOLO algorithm.

KEYWORDS: *Human Recognition; Motion Detection; Internet-of-Things-Based Motion Detection System; YOLO Algorithm; False Alarm Detection;*

1.0 INTRODUCTION

Security in a home or office is a critical aspect that the owner must consider seriously. This aspect helps the owner to secure assets like confidential documents, money, jewelry, and many objects inside the building. The owner could increase the security by hiring a security assistant or adding an Intrusion Detection Device or System like Closed Circuit Television (CCTV). An Intrusion Detection Device or System help the owner to identify if a person is trespassing at his house or office by activating the alarm and contacting the police at the same time. Furthermore, this device or system could identify if there is motion detection happening within the range. When the device detects a suspicious motion within the range, this event will trigger an alert to the owner. Usually, the owner receives the notification through a smartphone via specific apps like Telegram or SMS messages as shown in Figure 1.

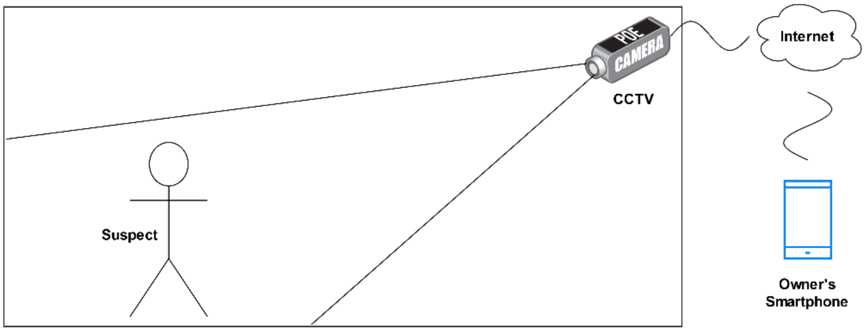

Figure 1: How Motion Detection Security Camera Detect and Alert The Owner

Due to the needs of security system, several studies have successfully developed a secure motion detection system that can recognize objects compared to conventional models. This study explains the previously proposed models in chronological order and their problems. The first model proposed by an article [1] in 2017, this model used an HD camera and OpenCV to detect motion within camera range. By using an HD Camera in Raspberry Pi, the system recognizes motion easier compared to a normal camera.

In the next year (2018), two articles [2], [3] proposed models for motion detection with human recognition features that detect face features with Haar-like features, Haar Training, Consensus-based Matching, and Tracking Keypoints (CMT). These models have better detection accuracy in many illumination conditions.

In 2019, studies focused on an embedded system like Raspberry Pi.

They implemented Face Recognition models to understand the algorithm's behaviour in low-end devices [4]–[7]. Some studies also found that the Raspberry Pi platform has enough computation power to handle face recognition and motion detection[8]. Besides that, Raspberry Pi is useful in space-constrained areas [9].

In recent years, motion detection and object recognition system is still being developed. Many studies have implemented numerous face recognition algorithms or methods inside an embedded system using Raspberry Pi. Some studies implemented many sensors to make precise detection [10]. Alternatively, some studies used Frame Differencing Method, Haar-Cascade algorithm to detect Human Movement[11] or OpenCV in a different programming language [12] to achieve the most optimal solution. And the most recent study added an Alert Mechanism to inform the owner when the device detected a motion [13], [14].

According to the previous proposed models, this study found two common problems in each model. The first problem is about how the device or system stores the detection data, and the second one is about the false alarm rate of the motion detection system. Most devices store their data in a hard drive or card storage system unencrypted form. Thus, the trespasser could take the memory away from the camera to remove the evidence. The alert means nothing if the evidence is not there. This problem will increase the investigation time, and the suspect will escape before the investigation finish.

The second problem is about the false alarm made by motion detection. This false alarm is caused by two possibilities, non-human movements (animal, trees, and wind) and digital camera defects [15]. An increasing number of false alarms could annoy the owner with a message sent by the device. Besides that, the device increased the storage with unnecessary recording. This problem could lead to a more dangerous situation where the owner distrusts the alert and let the situation happen.

Because of that, this study has the purpose a model for human face recognition by adding Human Motion Detection with YOLO Algorithm. The Microsoft COCO dataset has been used for training and testing. This study also adds a redundant Blockchain that stores exact data as the primary Blockchain to prevent further modification if a third party gains access to the primary one. Hence, this combination could reduce the false alarm rate made by the previous models and increase the data validity.

2.0 METHODOLOGY AND SYSTEM DESIGN

To increase the development effectiveness, this study uses the agile development method to develop the proposed model [16]. This method is appropriate for IoT model development that is prone to error during performance testing. Hence, the produced model has better feasibility compared to previous proposed models.

2.1 Hardware Configurations

Before developing the model, this study designs the electronic installations first. Because Raspberry Pi's models already provide a Camera Interface through the CSI interface, this study only needs to plug the flexible cable into that port. To give the camera the capability to capture at night, this study also adds two Infrared Torches on the Right and Left sides of Camera. The schematic of the proposed model is shown in Figure 2:

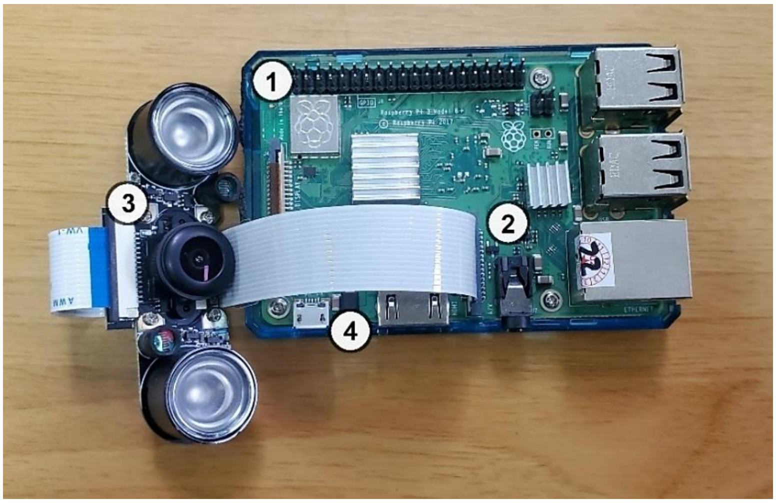

Figure 2: Fish-Eye Camera with Infrared Torches in Raspberry Pi 3B+ installed via CSI Interface

According to Figure 2, this study installed Fish-Eye Camera with two Infrared Torches (3) via CSI Interface (2) in Raspberry Pi 3B+ (1). After the requirement is completed, the power supply is ready to be plugged via a microUSB port (4).

2.2 Model's Process Flowchart

In this section, this study explains the system flowchart from motion detection with OpenCV via Frame Differencing and object recognition with YOLO. The COCO Dataset is used in this study for training and testing the YOLO algorithm to learn the object frame detected. Although the computing power is limited, Raspberry Pi can do some Image Processing sufficiently [17]. Frame differencing found the different contours between two frames. If there is a difference between the two frames, the model starts recognizing the object. Meanwhile, YOLO and COCO datasets are the core of object recognition in the proposed model. Both are known for fast [18] and accurate recognition [19]. Because of that, YOLO is suitable for real-time object recognition in many studies. The figure below explains the flowchart of the proposed model:

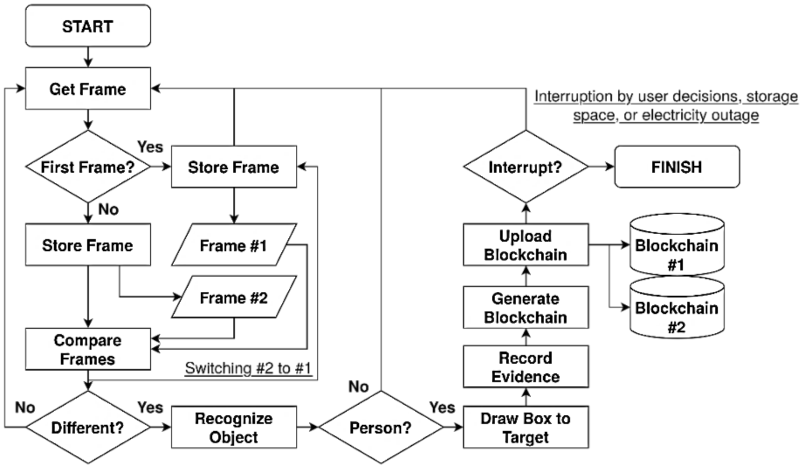

Figure 3: Flowchart Process inside the proposed model

According to Figure 3, the model must capture several images as frames to recognize motion and objects. When the model gets the first frame, it repeats the process to get the second frame. After the model gets two frames, then the comparison process begins. With the Frame Differencing method and OpenCV [20], the model could tell if motion occurs. If a difference is detected, the model tries to recognize what is inside the frame. If there is no person inside the frame, then the process repeated the capture step. If a person is detected, then the model continues to the next step. The model draws a box around the object, record the evidence, generate the blockchain, and upload it to two databases. These processes keep repeating until an interruption occur.

2.3 Network Topology and Database Connection

Network and Database setup is required in this proposed model. This configuration allows the model to communicate with the user via Telegram Bot and Realtime Database via Firebase service. The network topology tells the wireless setup and the communication process to both services (Telegram and Firebase). The network configuration of this model is shown in Figure 4:

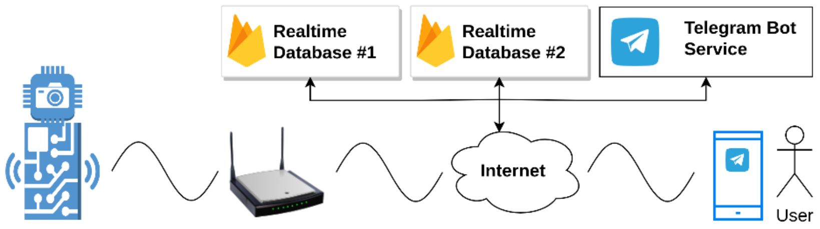

Figure 4: Network configuration to enable communication between the proposed model, services, and end-user

According to Figure 4, the proposed model connected via a wireless router to gain access to the Internet. When the model is connected to the Internet, the model uploads the data and notify the end-user via Telegram Bot. Hence, the user can receive the alert anytime and anywhere if the smartphone is connected to the Internet (whether Mobile Data or WiFi access). Besides that, the user can query the system to check or retrieve the data inside the Blockchain manually if required.

2.4 Evaluation Step

The last part of this section is to evaluate the proposed model. The first evaluation is via performance benchmark to understand computer resource usage during detection and recognition. This study tests the proposed model within 24 hours full without pause or stop (unless unexpected events occur). With this method, this study gathers many data as possible to find the most accurate data.

Besides that, the most effective way to measure the performance of the proposed model is to compare it side-by-side with the previous model. This study used the model from the article[14] as the base compared with the proposed model. By comparing the data side-by-side, this study can understand how the proposed behaviour detects and recognises humans in real situations.

3.0 RESULTS AND DISCUSSIONS

This section contains the implementation result, the benchmark evaluation, and the comparison result from the proposed model. The first explanation is about the recognition result when the model detects a human within the capture range. Figure 5 is the result of human recognition:

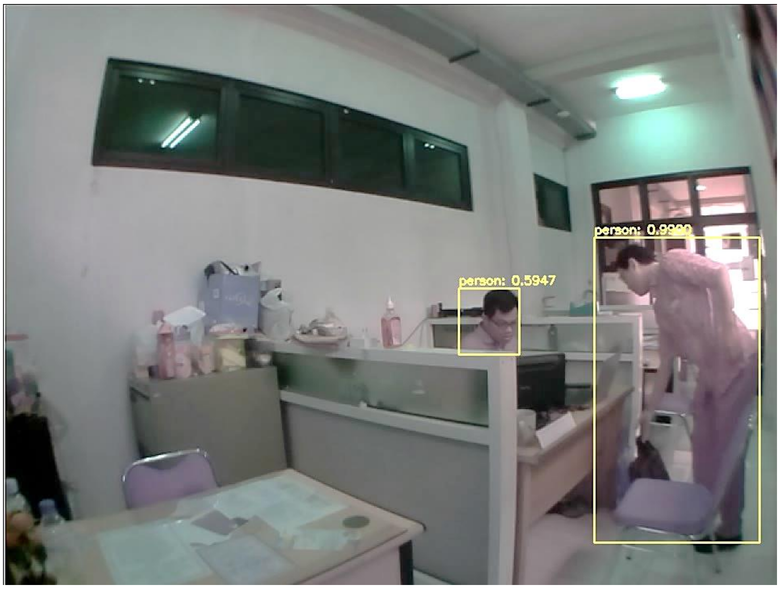

Figure 5: The proposed model detected persons inside the image and drew a box around them

According to Figure 5, the proposed model can differentiate between normal objects and persons. The figure showed two persons inside a box with the accuracy written above them. The accuracy usually increases if the algorithm detects a larger area of the object. The next result is the performance benchmark of the proposed model. This study obtained more than 500 rows of data for 24 hours course. Thus, this study decided to do data reduction on 15 minutes basis. After the data reduction process, this study obtained 97 rows of data that reflected 24 hours of activities. Table 1 contains data sample after reduction process:

Table 1: Sample benchmark data after the data reduction process

No	Time	CPU (100%)	Memory Data	Temp	Diff	Accuracy
1	1:00:00	53.975	103.363	60.148	1.111	0.000
2	1:15:00	52.269	110.960	68.084	4.167	0.590
3	1:30:00	51.108	110.509	71.267	13.704	0.529
4	1:45:00	51.019	123.281	72.926	7.917	0.000
5	2:00:00	53.681	127.715	73.733	9.583	0.000

Table 1 in the sample data contained several columns such as Detection Time (15 minutes basis), CPU usage (in 100% range), Memory Usage for Data (MB), Temperature (in Celcius), Difference (in percentage), and Confidence or Accuracy (in percentage).

The next step is to visualize the benchmark result in a diagram format according to its timeline. This study compiled the diagrams for Detection, CPU usage, Memory Data usage, and Temperature in a 2x2 grid layout in Figure 6:

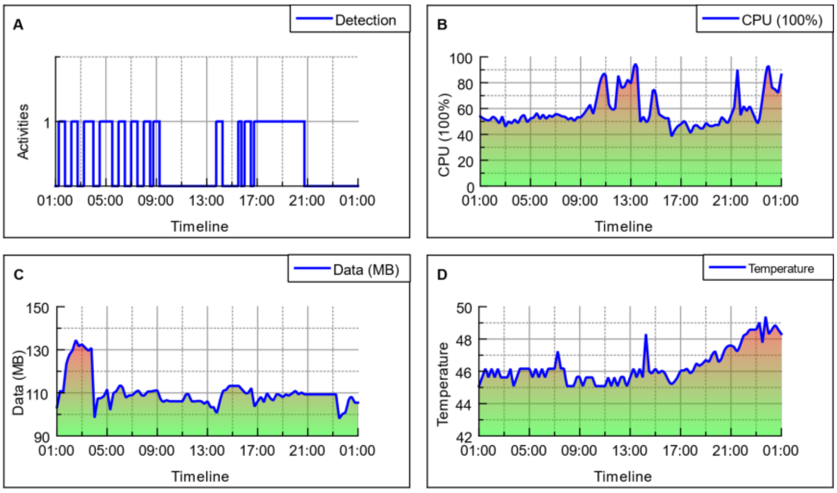

Figure 6: (A) The detection result, (B) The CPU usage, (C) The memory used, and (D) Temperature of the model

According to Figure 6, the proposed model produced a benchmark result, as shown in the Time-series diagram. Diagram (A) presents human activities within 24 hours span. Diagram (B) presents the CPU resources used by the model to detect, recognize, and record objects with an average of 58.362%. Diagram (C) presents the memory used to store camera frames temporarily. The model used an average of 110.123MB to do a single process. Diagram (D) is the board's temperature during the detection and recognition processes. The model's temperature reached an average of 42.33 degrees Celsius.

The next evaluation is a comparison between the previous and the proposed model. The previous model here means the motion detection without the YOLO algorithm. This study lineup the timeline of the benchmark to match the detection time. Since the previous model only had 33 rows, this study used the same data number to balance the comparison process. Figure 7 presents the motion activities comparison result between the previous and the proposed models:

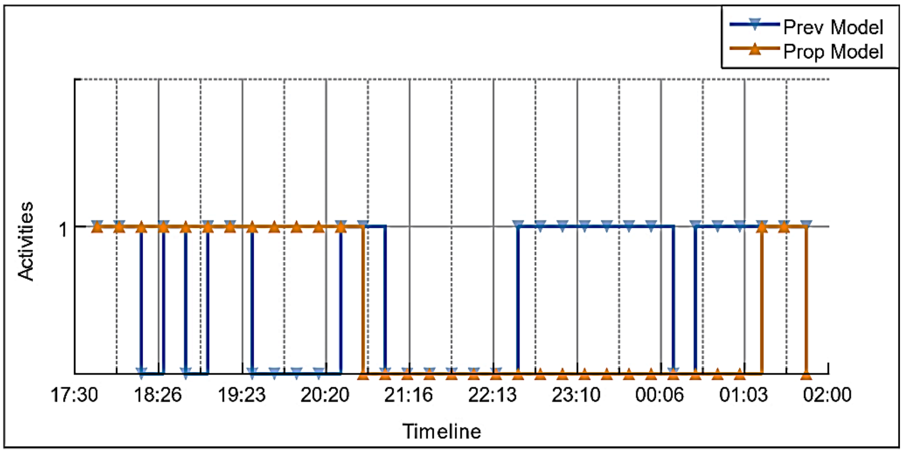

Figure 7: The motion activities detected by the previous (blue line) and the proposed model (orange line)

According to Figure 7, the previous model detected motion activities in as many as 20 activities. Meanwhile, the proposed model only detected 14 activities. The proposed model did not detect any human movement from 20.45 until 01.00. On the contrary, the previous model detected several motion activities in that time span. This proves false alarm indication caused by normal motion detection that cannot recognize human movement.

For this reason, this study concluded that the proposed model could detect motion caused by human activities and report the detection to the user via Telegram Bot. Furthermore, the user could request the system to validate both Blockchains through Telegram as well. In the performance aspect, YOLO algorithm and OpenCV have heavily loaded the computational operation to process many frames. However, Raspberry Pi 3B+ could handle the burden well. Need to be noted that high processing power produces high temperature. Hence, active cooling system is required.

4.0 CONCLUSION

This study came to the conclusion that the proposed model may reduce false alarms by detecting and recognizing the moving item based on the assessment that was done. If a person is the moving object, the model will accurately draw a box around the person. The user may therefore determine if a human is present or not. This study also assessed the model's performance to gauge its viability. The model utilized 110.123MB of memory and an average CPU usage of 58.362%. These applications still adhered to the Raspberry Pi 3B+ standard. Even if the board could manage the load efficiently, sufficient cooling system is still necessary to prevent processor overheating.

ACKNOWLEDGMENTS

This study is successful because of the guides, advices, and financial supports from *Lembaga Penelitian dan Pengabdian Kepada Masyarakat (LPPM) Universitas Semarang.*

REFERENCES

[1] Rashmi Mishra and Vandana Khare, ‘Low Cost HD Video Surveillance and Recording System using Raspberry Pi’, *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 7, issue-6S, pp. 22–25, 2019.

[2] P. Y. Kumbhar, Mohammad Attaullah, Shubham Dhere, Shivkumar Hipparagi, ‘Real Time Face Detection and Tracking using OpenCV’, *International Journal for Research in Emerging Science and Technology*, vol. 4, no. 4, pp. 39-43, 2017.

[3] W. S. Yuwono, D. Wisaksono Sudiharto, and C. W. Wijiutomo, 'Design and Implementation of Human Detection Feature on Surveillance Embedded IP Camera', in *2018* International Conference on Sustainable Information Engineering and Technology (SIET), Nov. 2018, pp. 42–47. doi: 10.1109/SIET.2018.8693180.

[4] Z. Balogh, M. Magdin, and G. Molnár, 'Motion detection and face recognition using raspberry pi, as a part of, the internet of things', *Acta Polytech. Hung.*, vol. 16, no. 3, pp. 167–185, 2019.

[5] Pranjal, N. Bishnoi, M. Kumari, R. Rishu, V. Shukla 'Object Detection and Tracking in Compressed Domain', *International Journal of Novel Research and Development,* vol.7, no 5, pp. 750–754, 2022.

[6] R. Mehta and S. Marodia, 'Human motion detection and notification system', *International Journal of Advance Research*, vol. 5, no 3, pp. 838-841, 2019.

[7] S. M. R, F. Khan, G. G. R, and H. S, 'Object Detection and Human Identification using Raspberry Pi', in 2019 1st International Conference on Advances in Information Technology (ICAIT), Jul. 2019, pp. 135–139. doi: 10.1109/ICAIT47043.2019.8987398.

[8] A. Ernest Rahul, 'Face Detection and Recognition System using Raspberry Pi', *International Journal of Research in Engineering, Science and Management,* vol. 1, no. 12, pp. 322–325, 2018.

[9] T. Won, B. Kim, S. Park, and J. Heo, 'An IoT Based Distortion Invariant Motion Detection Using Raspberry Pi with Mean Shift Algorithm', in Proceedings of the Conference on Research in Adaptive and Convergent Systems, New York, NY, USA, 2019, p. 63â€"64. doi: 10.1145/3338840.3355689.

[10] C. M. Srilakshmi and M.C. Padma, 'IoT based Smart Surveillance System', *International Research Journal of Engineering and Technology,* vol. 4, no. 5, pp. 2222–2224, 2017.

[11] Y. Erdani, A. F. Rifa'i, and A. Calfarera, 'Developing Human Movement Monitoring System using HAAR Cascade Classifier Algorithm', in 2021 3rd International Symposium on Material and Electrical Engineering Conference (ISMEE), Nov. 2021, pp. 253–256. doi: 10.1109/ISMEE54273.2021.9774239.

[12] 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), Feb. 2021, pp. 1378–1382. doi: 10.1109/ICICV50876.2021.9388404.

[13] M. D. N. Akash, CH. M. Kumar, G. B. Chakravorthy, and R. Aluvalu, 'Motion Detection and Alert System', in Intelligent Computing and Networking, Singapore, 2022, pp. 248–259.

[14] A. Hirzan, W. Adhiwibowo, and K. Gunata, 'Blockchain-based Motion

Detection in Smart Home with OpenCV and Raspberry PI', *Journal of Advanced Computing Technology and Application (JACTA)*, vol. 4, no. 1, May 2022, [Online]. Available: https://journal2.utem.edu.my/jacta/article/view/5247

[15] G. H. Goldman and M. Benyamin, 'Subpixel Motion Detection Using a Commercial Video Camera', *IEEE Sens. Lett.*, vol. 4, no. 3, pp. 1–4, Mar. 2020, doi: 10.1109/LSENS.2020.2977081.

[16] S. Alsaqqa, S. Sawalha, and H. Abdel-Nabi, 'Agile Software Development: Methodologies and Trends', *Int. J. Interact. Mob. Technol. IJIM*, vol. 14, no. 11, pp. 246–270, Jul. 2020, doi: 10.3991/ijim.v14i11.13269.

[17] H. K. Kondaveeti, D. Bandi, S. E. Mathe, S. Vappangi, and M. Subramanian, 'A Review of Image Processing Applications based on Raspberry-Pi', in *2022* 8th International Conference on Advanced Computing and Communication Systems (ICACCS), Mar. 2022, vol. 1, pp. 22–28. doi: 10.1109/ICACCS54159.2022.9784958.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, 'You Only Look Once: Unified, Real-Time Object Detection', in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[19] T.-Y. Lin *et al.*, 'Microsoft COCO: Common Objects in Context'. arXiv, 2014. doi: 10.48550/ARXIV.1405.0312.

[20] H. Singh, 'Advanced Image Processing Using OpenCV', in Practical Machine Learning and Image Processing: For Facial Recognition, Object Detection, and Pattern Recognition Using Python, H. Singh, Ed. Berkeley, CA: Apress, 2019, pp. 63–88. doi: 10.1007/978-1-4842-4149-3_4.