BLOCKCHAIN-BASED MOTION DETECTION IN SMART HOME WITH OPENCV AND RASPBERRY PI

Alauddin Maulana Hirzan[1], Whisnumurti Adhiwibowow[2] and Krida Pandu Gunata[3]

[1]Fakultas Teknologi Informasi dan Komunikasi,
Universitas Semarang, Tlogosari, 50196 Pedurungan, Semarang, Indonesia.
[2]Fakultas Teknologi Informasi dan Komunikasi,
Universitas Semarang, Tlogosari, 50196 Pedurungan, Semarang, Indonesia.
[3]Fakultas Teknologi Informasi dan Komunikasi,
Universitas Semarang, Tlogosari, 50196 Pedurungan, Semarang, Indonesia.

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

Article History: Received 10 Mac 2022; Revised 20 April 2022; Accepted 30 May 2022

ABSTRACT: Home security is a serious problem that must be compromised. A simple mistake like forgetting to lock the door can result in tremendous material losses. There are several ways to help the homeowner mitigate the risk. One of them is by implementing a motion detection system. This model can help the homeowner detect unknown movement in a vacant house. Besides that, the recording from the sensor can help the investigation process. However, the current model had a problem where the data were modifiable. Thus, modified data could slow down the police investigation. This study aims to improve a computer vision-based motion detection model with blockchain-based that serves as end-to-end security. With this improvement, the model can detect invalidity. Besides data protection, this model also provided Telegram access for the end-users. According to the evaluation result, the proposed model successfully captured 300 recordings with a 98% success rate. In terms of performance, the model needed up to 42% of CPU in a multiprocessor environment. The memory usage for the process needed 12.94 MB, and the data usage was 107.18 MB on average. These facts concluded that the proposed model provided data protection with a good performance aspect for embedded systems.

KEYWORDS: *Blockchain, Computer Vision, Internet of Things, Motion Detection, Telegram Bot*

1.0 INTRODUCTION

A household is usually protected with a door and window lock to prevent theft or burglary. This type of security is commonly used everywhere in the modern era. Whether front door or garage, the homeowner uses this whenever leaving the house. However, this does not mean that security will be secure forever. The thief also adapts the method they use to enter illegally. Thus, the homeowner must think of another way to secure their house. Many modern households use a technology-based lock to improve their security. They used simple technology like tripwire alarms or an advanced model of the face recognition system. One form of monitoring in modern households is motion detection. Unlike the face recognition model, the motion sensor will record a video triggered by a motion detected by the camera[1]. This model stores the evidence in the form of a video where the homeowner can retrieve it for monitoring. If it turns out that something serious has happened, such as a robbery or house fire, the homeowner can immediately report it to the authorities. Figure 1 below explains how motion detection works.

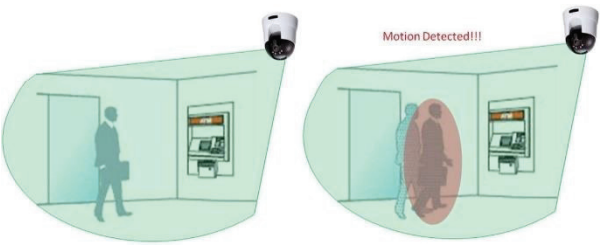

Figure 1: Motion detection will detect any motion within sight[2]

As shown in Figure 1, the model used computer vision as the motion sensor. Although this solution needs expensive computation, the coverage for the detection is wider than the PIR sensor. This model works by comparing one frame with the next frame. If there is difference between them, the model will trigger the specified action.
Several studies have been done before to improve how motion detection work. Some studies relied on a cheap solution with a PIR sensor, and the rest relied on Open Computer Vision (OpenCV) solution. In 2017, the study found that Raspberry Pi was used for motion detection with low frame rate. With low frame rates up to 4-5 fps with marker and 8 fps without marker, the study confirmed that Raspberry Pi could be used as motion detection[3]. In the next study of the same year, the model improved with Harr Cascade classifier for better detection. Unlike previous model that only relied on OpenCV, this model could detect motion and face[4]. In the next article of 2018, the study abouot motion detection continues. The model in the study

has been improved with reporting capability through e-mail. Thus, any event that occurred within coverage will be reported immediately to the owner[5]. However, there are several problems with e-mail as reporting media. The most common problem with e-mail is the direction of the communication. When using e-mail, the owner cannot send a command to the device. Thus, in the next study Telegram was proposed to replace e-mail. In the recent study, Telegram was used as the reporting media for the motion detection model. With the technology of Telegram Bot, the owner can send commands or receive reports from the device anywhere and anytime[6], [7].

According to the previous studies above, this study found common problems with the overall models. The first problem that this study found is about the security of the data[8]. All the existing models proposed a security model for the smart house, but they forgot about the data protections. Thus, the third party can manipulate or destroy the evidence. The second problem is about how the previous studies test their models. Most of the models only cover the functionality of the model. None of them are talking about the performance in long term. Thus, the feasibility of the device on longer period is questionable.

Due to these reasons, this study proposed a Blockchain-based Motion Detection Model with Raspberry Pi and OpenCV. Similar to previous models, the proposed model uses OpenCV to determine whether something is moving within the frame or not. However, this time this study implements a Blockchain-based data storage mechanism to solve the first problem. The blockchain stores the metadata of the captured data and the identifier of the file. Hence, the end-users can check the validity of the data and the video itself. To ensure the feasibility of the model in a real situation, this study will conduct the evaluation in room within a specified period.

2.0 METHODOLOGY AND SYSTEM DESIGN

To design the system, this study uses the Agile development methodology. This type of methodology relied on integration and testing before moving forward to the next development step. Hence, increasing the effectivity of the development.

Agile development method[9] is known within software development to help increase the effectiveness of the development process. Unlike other methods, this method relied upon integration and testing in every development step. Hence, the development will not progress unless the previous step success. The system design consists of several parts like hardware schematic, camera control, blockchain, database, topology, and evaluation.

2.1 Hardware Schematic

The first step is to lay out the specification of the model. Since the model only uses a camera as the sensor, the schematic of the installation is simple. Most Raspberry Pi devices (except for the Pico series) are equipped with a camera interface. Thus, plugging the camera cable is already enough.

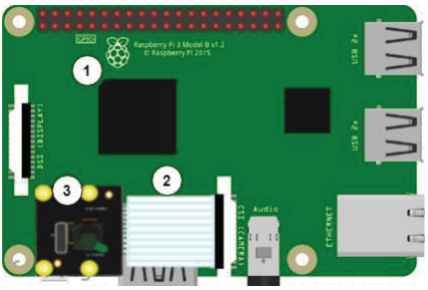

Figure 2: The schematic installation of CSI camera

According to Figure 2, the installation of the night vision camera (3) is quite straightforward. By connecting the Flat Flexible Cable (2) to the CSI port, the Raspberry Pi (1) can access the camera through its application. When everything is normal, the camera will turn on its LED. If not, then the camera might need readjustment. After this phase is complete, this study can move to the next step.

2.2 System Design

Controlling the camera is also the main aspect of the model. The camera controller contains the most important function for the motion detection mechanics, where the model compare two frames. Figure 3 below explains how motion detection works with camera's frames.

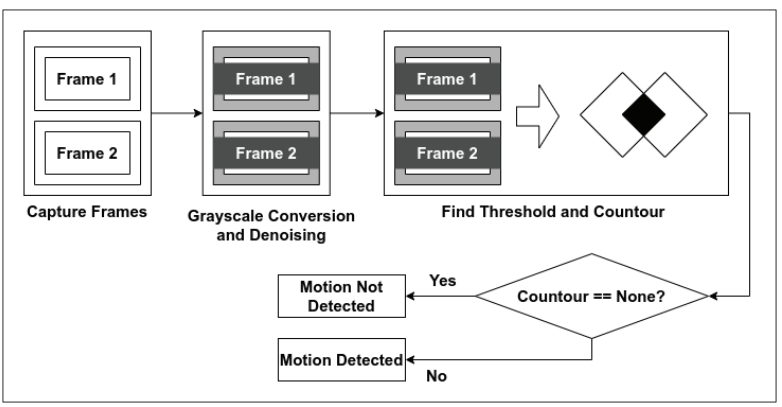

Figure 3: The comparison process between two frames

Figure 3 explains the process of comparing two frames from the camera with OpenCV[10]. Before comparing the frames, the model must preprocess the frame into better data. The first step of data preprocessing is to convert the colour into grayscale. This step helps the comparison process by reducing the colour into grayscale and enhance the feature inside the frames[11]. After that, the frames must enter the denoising process to remove grains within the frames. Next the model can compare the frames by finding the threshold and the countour between them[12]. If the countour returns None, then the frames are identical. If not, then the countour will return the intersect part. This is the indicator if something is moved or not.

Due to the processing capability of Raspberry Pi, this study will not use any sophisticated Blockchain algorithm available in the market[13], [14]. Hence, this study designs a basic Blockchain template that is lighter and easier to use.

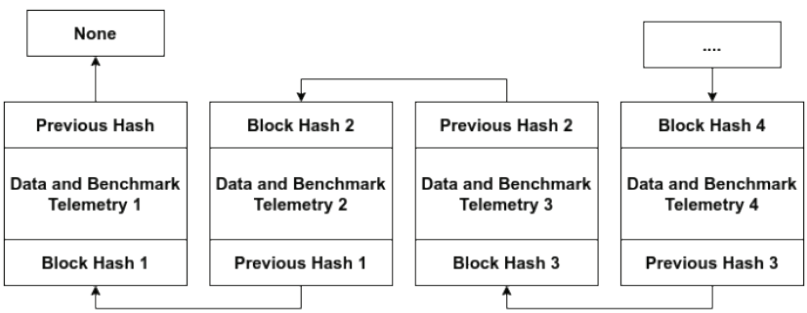

Figure 4: The Blockchain Design for Motion Detection

The Blockchain design in Figure 4 consists of many blocks and chains. Each block also consists of three different parts. The first part of the block contains a checksum hash from the previous block (If this block is the first, then this part contains None). The second part contains the video identifier and telemetry for benchmark. And the third part contains the checksum hash of the current block[15], [16]. The block is indicated as invalid when the previous hash is not identical. Thus, the data may be invalid, or interruption has occurred[17], [18].

The next step is to design the upload mechanism to the database. This study uses a cloud service like Firebase Realtime database and Google Storage. The mechanism of uploading data is shown in the following figure 5.

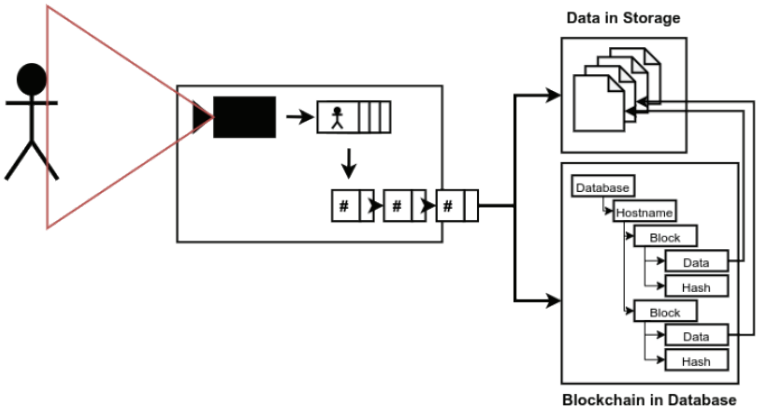

Figure 5: Upload Mechanism for Blockchain and Recording

The upload mechanism in this proposed model uses a different approach, especially when storing recording blobs. According to Figure 5, the recorded motion video blob and its encrypted metadata are stored in different places on the Google platform. This study decided to do this because Firebase Real-time Database limits the data up to 1024 bytes. Hence, the database is not suitable for larger data. As a countermeasure to this problem, Google provided an additional feature to store large blobs in Google Storage.

2.3 Topology and Evaluation Setup

In this section, this study will explain how the configuration for the network and the evaluation. The network topology configures the connectivity between devices from the camera and clouds. Besides that, the topology also covers the connectivity to Telegram Bot and the end-users.

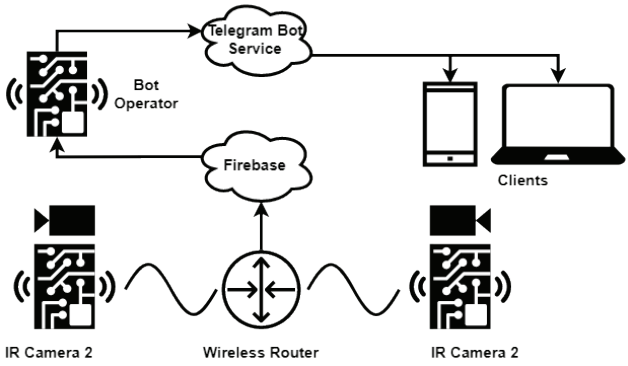

Figure 6: The network topology for the proposed model

The topology in Figure 6 consists of several devices like two Raspberry PIs that act as Motion Sensor, a wireless router for cameras, one Raspberry Pi for operating Telegram Bot, and a client (either laptop or smartphone with Telegram App installed). Both cameras will send encrypted blocks through wireless router and stored in Firebase. Both cameras will send encrypted blocks through a wireless router and store them in Firebase. Meanwhile, the Telegram Bot operator is on standby and ready to receive any query from the end-user. When the end-user sends a command to the Bot, the operator will fetch the data from the database and send them through Telegram API[19]. Hence, the end-user can receive the data anytime and anywhere[20].

The last step for the proposed model is the evaluation phase. Unlike previous studies that only test the models for a short time, this study will test the model for a longer period. This study places each camera in the corner of a square-shaped room and leave the device active for several hours. To obtain the benchmark data, this study adds telemetry code to the Blockchain such as CPU usage, memory usage, data size, and execution time.

3.0 RESULTS AND DISCUSSION

In this section, this study will explain the result of the proposed model physically, the command list used to validate and retrieve the data, and the benchmark result from telemetry data. The following Figure is the physical device of the Infrared Camera installed in the corner of a room:

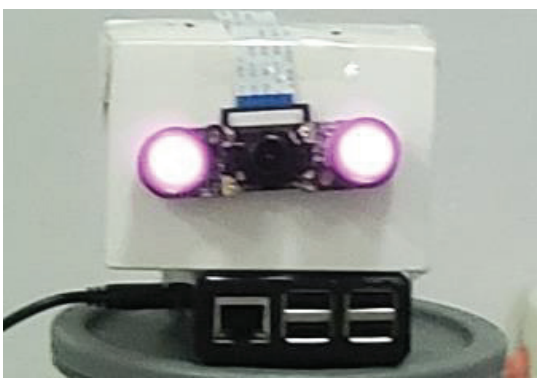

Figure 7: Physical IR Camera with Raspberry Pi

The IR Camera in Figure 7 is a motion detection model with OpenCV. The model is connected to a wireless network, so it can upload the recording when a motion detected within its field. Since the camera is equipped with Infrared torches, it can see well at night.

The cameras uploaded all the recordings to the Cloud and only the Telegram Bot operator can retrieve them safely. The following figure contains available commands in the Telegram Bot.

Figure 8: Available Commands in the Telegram Bot

The operator of the Telegram Bot is configured to receive several commands as shown in Figure 8. There are three types of commands available: all data, selected data, and validation queries. The command with the type "All Data" fetches necessary information from all blocks in the Blockchain. The "Selected Data" type fetches specific information from a block or machine. The validations command type is used for validating information inside the Blockchain. The following Figure shown the result of data retrieval:

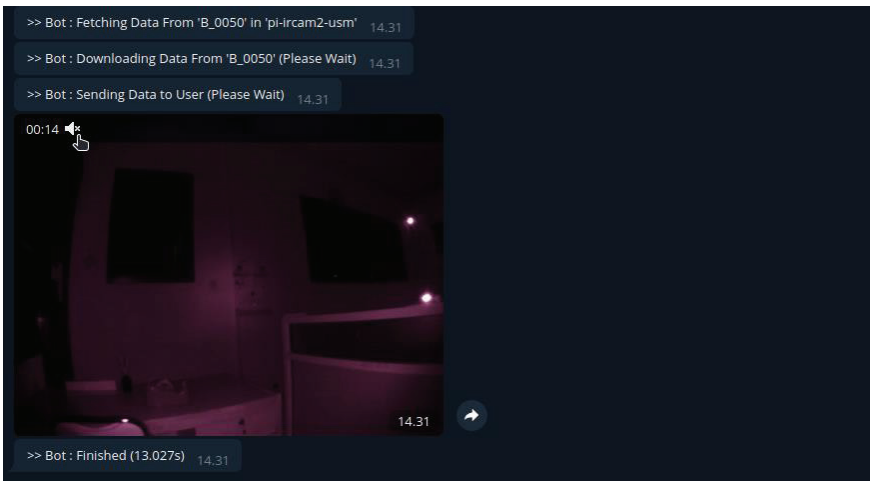

Figure 9: Retrieving video via Telegram

With a "/data <block> <machine>" command, this study can retrieve a video from Firebase Real-time Database with Telegram Bot. The operator will fetch the data and send it to the end-users. So, the end-users can watch the video with their Telegram Apps. Figure 9 showed the successful result of the command. The bot sends a video where the user can monitor what happened in that time.

As this study stated before, this study evaluates the proposed model within several hours. This study has activated our devices for eight hours and successfully captured total 300 recordings from two cameras. According to the validity check through Telegram, the bot found five broken blocks inside the Block chains. Thus, only 98% of the total blocks remained valid (295 blocks). After further analysis, this study found that the network timeout occurred during eight hours of testing. A slow and unreliable wireless network is one of many external factors that need to be carefully configured.

The next result is about the activity of the motion detection. Since the recordings are scattered within the timeline, this study has grouped the recordings for every 15 minutes. By grouping the recording, this study can easily determine when the motion detection is active. The following Figures 10 and 11 show the motion activities within eight hours:

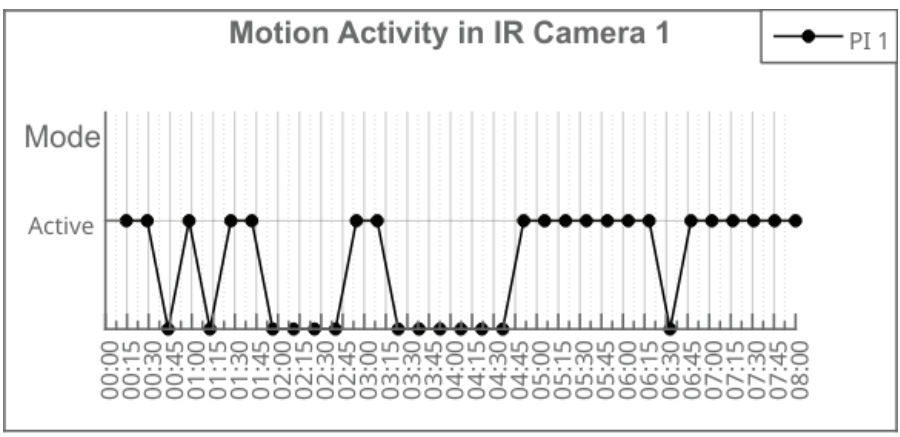

Figure 10: Motion Activities in IR Camera 1

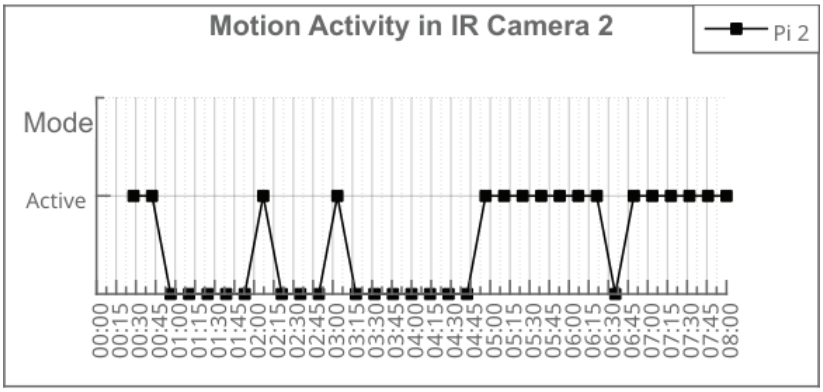

Figure 11: Motion Activities in IR Camera 2

Figures 10 and 11 illustrate the motion activities that occurred within eight hours. The graph result in Figure 10, the first camera detected the motion 20 times. In Figure 11, the second camera detects 17 motions. Based on these results, the motion detection cameras successfully detected the motion within the specified time.

The last part of the result is the benchmark result. The following table is the average result of the benchmark:

Table 2: The average of the benchmark result for each device

No	Device	CPU (400%)	CPU (100%)	Memory	Data	Estimate (s)
1	pi-ircam1-usm	172.82%	43.2%	12.96	107.41	73.876
2	pi-ircam2-usm	166.76%	41.69%	12.92	106.95	71.189
Average			42%	12,94	107,18	72.532

The average benchmark result in Table 2 shows our evaluation results. This study found that both models used more than 150% of the CPU to process the whole process. The first model used around 172.82% and the second model used around 166.76% (in 400% scale). However, these results are obtained through Python libraries. Thus, the results might be more than 100% if the process uses more than one processor. Since the results are not feasible, this study scales down to a 100% scale the result according to the number of logical processors in each system. If each system has four logical processors with total range up to 400%, then the final usage for this model is 43.2% for the first model and 41.69% for the second model. The total average for all machines is 42%.

The next parameter is the memory size. This parameter shows how much memory the program needs to do the whole process. According to our benchmark, both models used the same size of memory. Besides

memory size, this study included Data size to show the data usage of each model. The total average for memory usage is 12.94MB. The results are quite high compared to the previous parameter since Data Size depends on the Data inside the model. According to the table, the first model used around 107.41MB and the second model used around 106.95MB. The total average of data size is 107.18MB.

The last parameter is the required time to do the comparison, recording, and uploading processes. The benchmark showed that the first model required 73.876s and the second model required 71.189s to run the repeating processes. Thus, on average all machines need 72.532s to execute the task.

According to the overall result of the evaluation, this study can conclude that the proposed model can detect and upload any recorded motion into clouds within eight hours. Besides that, the operator of Telegram Bot also successfully retrieved and sent the recording to the end-users.

4.0 CONCLUSION

Many motion detection models have been proposed, many of them trying to improve the model by adding many algorithms or reporting mechanisms. All models were good for the security feature of Smart Home. However, there were many problems with the previous models. One important problem that needs to be addressed is the security of the data inside the model. Due to this reason, this study has the purpose to improve the data by providing a simple Blockchain mechanism to protect the validity of the data. According to the evaluation results, the models successfully detect records and upload encrypted data into the clouds in the shape of a block. Based on the validity test, the Blockchains contained five broken blocks and 295 valid blocks. However, this problem is caused by network timeouts. In terms of benchmark, the average CPU usage for both devices are 169.79% or 42% after scaling down to a 100% scale. The average memory is 12.94MB and the average data size is 107.18MB. The average execution time for the whole process in both devices is 72.532s for every data block. According to these results, this study concluded that the motion detection model with OpenCV and Blockchain successfully detects the motion and secures the recorded data.

ACKNOWLEDGMENTS

This study would like to thank Lembaga Penelitian dan Pengabdian kepada Masyarakat (LPPM) Universitas Semarang for the guide and financial support.

REFERENCES

[1] Z. G. Faisal, M. S. Hussein, and A. M. Abood, "Design and realization of motion detector system for house security," *TELKOMNIKA Telecommun. Comput. Electron. Control*, vol. 17, no. 6, p. 3211, Dec. 2019, doi: 10.12928/telkomnika.v17i6.13142.

[2] D. Battini, "Motion Detection with Open CV and C#," *Motion Detection with Open CV and C#*, 2018.

[3] Z. Balogh, Å. tefan Koprda, and M. TurÄ Ã¡ni, "Motion Detection Using HD Camera of Microcomputer Raspberry Pi," in *2017 IEEE 11th International Conference on Application of Information and Communication Technologies (AICT)*, Sep. 2017, pp. 1–6. doi: 10.1109/ICAICT.2017.8686849.

[4] K. N. K. Kumar, H. Natraj, and T. P. Jacob, "Motion activated security camera using raspberry Pi," in *2017 International Conference on Communication and Signal Processing (ICCSP)*, Apr. 2017, pp. 1598–1601. doi: 10.1109/ICCSP.2017.8286658.

[5] G. Muruti, F. A. Rahim, and Md. N. A. Zawawi, "Motion Activated Security Camera Using Raspberry Pi: An IoT Solution for Room Security," *Adv. Sci. Lett.*, vol. 24, no. 3, pp. 1698–1701, 2018, doi: doi:10.1166/asl.2018.11140.

[6] N. Hema and J. Yadav, "Secure Home Entry Using Raspberry Pi with Notification via Telegram," in *2020 6th International Conference on Signal Processing and Communication (ICSC)*, Mar. 2020, pp. 211–215. doi: 10.1109/ICSC48311.2020.9182778.

[7] A. H. Azhar, M. F. I. Othman, N. Bahaman, M. Z. Mas'ud, and Z. Sa'aya, "Implementation of Home Security Motion Detector using Raspberry Pi and PIR Sensor," *J. Adv. Comput. Technol. Appl. JACTA*, vol. 3, no. 2, pp. 42–51, Dec. 2021.

[8] I. Butun, P. Osterberg, and H. Song, "Security of the Internet of Things: Vulnerabilities, Attacks, and Countermeasures," *IEEE Commun. Surv. Tutor.*, vol. 22, no. 1, pp. 616–644, 2020, doi: 10.1109/COMST.2019.2953364.

[9] 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.

[10] G. Arva and T. Fryza, "Embedded video processing on Raspberry Pi," in *2017 27th International Conference Radioelektronika (RADIOELEKTRONIKA)*, Apr. 2017, pp. 1–4. doi: 10.1109/RADIOELEK.2017.7937598.

[11] 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.

[12] V. K. Dilip, H. Huang, A. Garg, X. Huang, and G. Mei, "A Novel Python Video Processing Program for Identifying and Quantification of Soil Cracks in Real Time," in *Proceedings of the 2nd International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications*, Singapore, 2022, pp. 39–46.

[13] D. Hawthorne, M. Kapralos, R. W. Blaine, and S. J. Matthews, "Evaluating Cryptographic Performance of Raspberry Pi Clusters," in *2020 IEEE High Performance Extreme Computing Conference (HPEC)*, Sep. 2020, pp. 1–9. doi: 10.1109/HPEC43674.2020.9286247.

[14] N. Alipour, M. Wang, and D. Krishnamurthy, "Oasis: Performance Matching IoT System Emulation," in *2021 IEEE 14th International Conference on Cloud Computing (CLOUD)*, Sep. 2021, pp. 378–386. doi: 10.1109/CLOUD53861.2021.00051.

[15] L. D. Xu, Y. Lu, and L. Li, "Embedding Blockchain Technology into IoT for Security: A Survey," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10452–10473, Jul. 2021, doi: 10.1109/JIOT.2021.3060508.

[16] T.-V. Le and C.-L. Hsu, "A Systematic Literature Review of Blockchain Technology: Security Properties, Applications and Challenges," *J. Internet Technol.*, vol. 22, no. 4, p. 14, 2021.

[17] M. El-Masri and E. M. A. Hussain, "Blockchain as a mean to secure Internet of Things ecosystems – a systematic literature review," *J. Enterp. Inf. Manag.*, vol. 34, no. 5, pp. 1371–1405, Jan. 2021, doi: 10.1108/JEIM-12-2020-0533.

[18] Y. P. Tsang, C. H. Wu, W. H. Ip, and W.-L. Shiau, "Exploring the intellectual cores of the blockchain–Internet of Things (BIoT)," *J. Enterp. Inf. Manag.*, vol. 34, no. 5, pp. 1287–1317, Jan. 2021, doi: 10.1108/JEIM-10-2020-0395.

[19] R. Parlika and A. Pratama, "The Online Test application uses Telegram Bots Version 1.0," *J. Phys. Conf. Ser.*, vol. 1569, p. 022042, Jul. 2020, doi: 10.1088/1742-6596/1569/2/022042.

[20] A. Trirahma, "Telegram Bot as a Data Collection Tool for Progress Reports in Area Mapping Progress Monitoring System | Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)," Dec. 2021, Accessed: Jun. 19, 2022. [Online]. Available: http://jurnal.iaii.or.id/index.php/RESTI/article/view/3654.