Scientific Journal of Informatics
Vol. 9, No. 2, Nov 2022

p-ISSN 2407-7658 http://journal.unnes.ac.id/nju/index.php/sji e-ISSN 2460-0040

The Development of Chicken Coop Automatic Remote Visual Monitoring System

Sri Wahjuni[1*], Suryo H. Sanjiwo[2], Wulandari[3], Auriza R. Akbar[4]
[1,2,3,4]Department of Computer Science, Faculty of Mathematics and Natural Sciences,
IPB University, Indonesia

Abstract.
Purpose: A remote visual monitoring system will be very helpful for chicken farmers to monitor their cage, which is usually located away from their houses. This system needs adequate bandwidth to transmit the video over the internet, which is usually very limited in urban areas. This research aims to develop an automatic chicken coop remote monitoring system and define the optimum video resolution to be transmitted.
Methods: We used an 8 MP Raspberry Pi V2 to record the video and send the results to Google Drive using the GDrive API. Furthermore, a live streaming video from the chicken coop is accessible through a simple HTTP web page utilizing ngrok as tunneling software. The live-streaming video can be publicly accessed anywhere using a web browser. Three video resolutions of 640x480, 800x600, and 1024x768 with 15 and 30 frame rates were used in our experiments. Each scenario has a duration of five minutes and takes 12 times.
Result: The experiment results showed that resolutions that provide stable video recording and streaming are 640x480 and 800x600. The resulting system succeeded in performing live streaming along with data acquisition.
Value: The Google Drive infrastructure is used because of its popularity and convenience by people with limited digital literacy, such as smallholder chicken farmers. Furthermore, the video produced by this system can support research on chicken behavior pattern identification to build a system notification of an emergency situation in the cage.

Keywords: Automatic Remote Monitoring System, Chicken Behavior, Google Drive, Ngrok, Video Streaming
Received January 2022 / **Revised** March 2022 / **Accepted** October 2022

INTRODUCTION

Based on livestock statistics, chicken farms contributed 62.3% of broiler meat production in national meat production in 2017 and a contribution of 81.85% of total poultry meat production in 2018 [1]. Indonesia's chicken meat consumption in 2020 is 3,275.3 (thousand tons), which increased 17.56% from the previous year [2]. The high demand for chicken meat consumption encourages farmers to make maximum efforts to increase the productivity and quality of their livestock [3]. One of the factors that affect chicken productivity is feed parameters [4]. In the opinion of [5] and [6], feed cost in a livestock business is the highest, ranging from 60% to 80%. The cost of broiler feed reached 75.45% of the processed primary data in 2016 [7]. The cost will increase if farmers' operational and travel expenses are added. The farmers must frequently check the coop to give the chicken food or check whether there is an unhealthy chicken. The sick chicken needs to be treated as soon as possible to prevent spreading the disease [8]. Chicken's condition has the possibility to threaten food safety if the infected meat is consumed [9], [10].

In general, the traditional way to monitor the behavior of chickens is to send farmers, officers, or breeders who come to the chicken coop to monitor and assist regularly, which is labor-intensive and time-consuming. The aforementioned manual monitoring system requires a relatively large amount of effort and time, especially if the farmer's house and cage are located quite far apart [11]. Furthermore, human visitation in a broiler house poses an infection risk to the farm [12]. Regarding that problems, an automatic remote visual monitoring system can be implemented to overcome the disadvantages of the manual method. Thus, visual supervision of chicken behavior, including food conditions and chicken health, can be done periodically without visiting the poultry farm's location. The previous research [13] was conducted to monitor the

*Corresponding author.
Email addresses: my_juni04@apps.ipb.ac.id (Wahjuni), suryo_jimbo@apps.ipb.ac.id (Sanjiwo), wulandari.ilkom@apps.ipb.ac.id (Wulandari), auriza@apps.ipb.ac.id (Akbar)
DOI: 10.15294/sji.v9i2.34630

locomotion behavior of broiler chickens in the cage. Overall, using a monitoring system increases farmer time efficiency and maintenance costs [14].

However, implementing a remote visual monitoring system needs adequate bandwidth to transmit the video over the internet. An inadequate poultry farm location network infrastructure may lead to slow or hampered monitoring data transmission. One alternative to overcome the problem of data transmission and also optimize the data transmission process is to define an optimal video resolution value. Furthermore, providing a streaming server supporting their real-time coop monitoring system is difficult for smallholder chicken farmers. Thus, this study focuses on building an automatic remote monitoring system using an 8MP Raspberry Pi v2 camera with a Raspberry Pi 3B and defining the optimum video resolution to be transmitted. The Google Drive cloud infrastructure is used because of its popularity and convenience by people with limited digital literacy, such as smallholder chicken farmers.

METHODS

This research aims to develop an automatic visual remote monitoring system using Raspberry Pi that is integrated with a camera for video capture. We started the study by analyzing the system requirements, such as the software and hardware requirements. Then we design the system architecture to meet those requirements. In order to carry out the two tasks, four steps are implemented in this research: the development of an application program for Raspberry Pi, the GDrive API settings, the ngrok settings, and the system implementation and testing. The experiment's results were analyzed to obtain the most optimum parameter recommended for an automatic remote visual monitoring system. The parameters involved in this analysis are file size, image resolution, running time, and uploading duration.

System Architecture

Figure 1 shows the system architecture developed in this research. The hardware consisted of a V2 Raspberry Pi 8MP camera, a memory card, and a Raspberry Pi 3B. The software used in this research is the Raspbian Stretch operating system, ngrok tunneling service 2.3, and Python CLI 3.4. We utilized Google Drive as storage media. A Raspberry Pi-based system architecture is designed in this study to handle the data/video acquisition as well as stream the video live. The recording format used in this study is .h264, which was later converted to .mp4. The Raspberry Pi streams the .mp4 file to the ngrok domain; meanwhile, at the same time, it sends the .mp4 file copy to Google Drive.

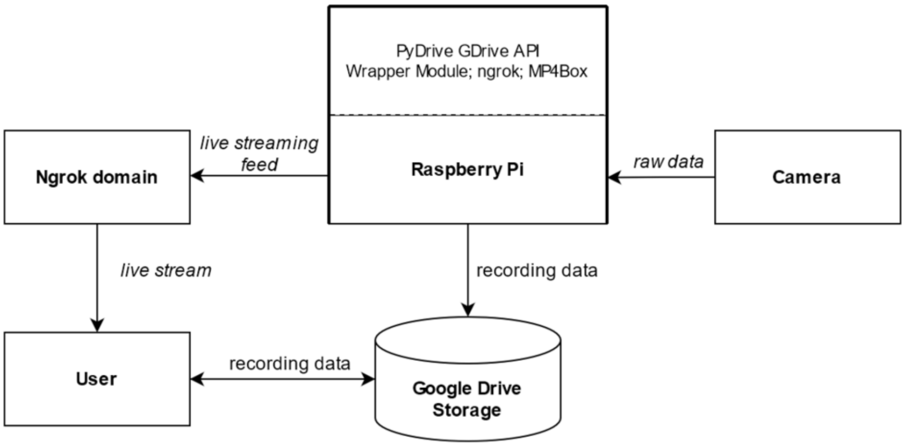

Figure 1. The system architecture

The Development of Application Program for Raspberry Pi

The Python programming language is used to write programs on the Raspberry Pi. However, commands via the Command Line Interface (CLI) can be used to operate the camera connected to the Raspberry Pi. The PiCamera module in Python is used to simplify the integration of camera operations with Python-based systems [15]. The monitoring system implemented the streaming feature that can be accessed from desktop and mobile web browsers, as well as video recording that will be stored in Google Drive using GDrive API. The PiCamera camera module will produce a video in .h264 format. The .h264-formatted video will be converted into .mp4 format by using the MP4Box tool to view the video in Google Drive. The MP4Box is a tool that can be used to convert various video formats into .mp4 format [16].

This streaming feature is implemented using the *http.server* module from Python, Motion JPEG (MJPEG) as the streaming base, and is presented in a basic HTML page. Motion Joint Photographic Expert Group or MJPEG is a format video compression where each video frame is processed and compressed separately to form a sequence of JPEG images. Each JPEG frame can then be displayed sequentially so that it looks like a moving video. MJPEG is often used in streaming functions because of its advantage of not draining system memory excessively [17]. This format is also used in research of monitoring systems using Raspberry Pi [18] and the study of intelligent camera systems [19].

The threading module in Python is implemented in this system to take advantage of the multithreading process to run video recording and streaming features simultaneously. The duration of the upload, the size of the uploaded video file, the conversion time of the video file, and the duration of the system run will be calculated when the system is started until the upload process is completed. These data will be analyzed to compare the performance of scenario testing. All features and the entire program will be integrated into one automated Raspberry Pi system.

The GDrive API Settings

The PyDrive is used as a wrapper to make it easier to use the GDrive API [20]. Some initial steps must be done prior to implementing the PyDrive/GDrive API. First, we create the Google API project at https://console.cloud.google.com, enabling the GDrive API and basic project configuration. Thus, we download the JSON configuration file containing the client ID that will be used as the GDrive API authentication method. Lastly, we execute the PyDrive quickstart program as initial authentication, which only needs to be done once to save the authentication .txt file in order to automate the following authentication process. The quickstart process flow can be seen in Figure 2.

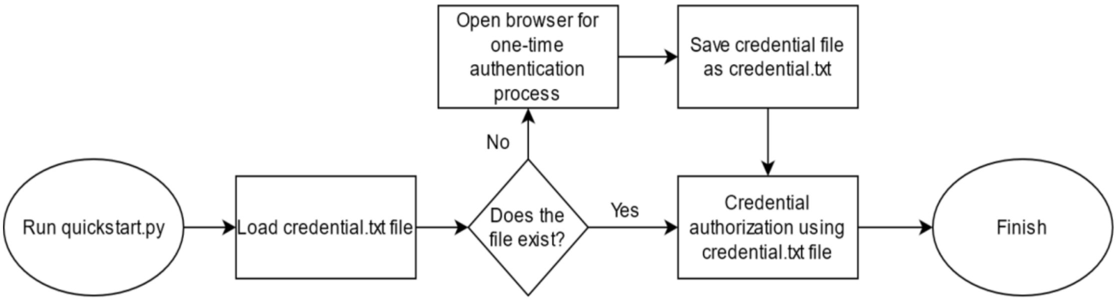
Open browser for one-time authentication process
Save credential file as credential.txt
No
Run quickstart.py
Load credential.txt file
Does the file exist?
Yes
Credential authorization using credential.txt file
Finish

Figure 2. Quickstart PyDrive process

The Ngrok Settings

Tunneling software ngrok can provide a subdomain of the ngrok.com domain as a form of remote port forwarding to publicly access local applications or web servers [21]. In this study, ngrok is implemented to forward streaming services based on the http protocol. Thus, ngrok is used to create a public domain to access streaming services. Streaming service can be accessed on the domain listed in the Forwarding section of ngrok when the domain is accessed via a web browser, and then camera streaming from the Raspberry Pi hardware can be viewed on a basic HTML page.

To be able to use ngrok, there are several steps to be taken. Firstly, we sign up for an account at https://dashboard.ngrok.com/signup. Then, we download the ngrok application according to the operating system at https://ngrok.com/download; in this study, the Linux version ngrok is used. After that, the downloaded ngrok application file in the .zip form is extracted. The last step is authenticating the ngrok application with the token obtained from https://dashboard.ngrok.com/get-started/your-authtoken; this process is only done once.

The System Implementation and Testing

The implementation and system testing scenario used is recording the video with a duration of five minutes 12 times for each resolution and different frame rates or frames per second (fps). Three resolutions and two fps variants were tested, 640x480, 800x600, and 1024x768, with 15 fps and 30 fps, respectively. The three resolutions were chosen because they are standard resolutions for image processing [22], [23]. The two fps variants in the test scenario were selected because 15 fps and 30 fps are common fps choices in monitoring systems [24] and in object detection research [25]. The system's flow is divided into five stages which can be seen in Figure 3.

Converting the .h264 to .mp4 video format is done by calling the CLI command in the Python program. The process of storing video parameter data in .csv form is done using Python's CSV module. The video parameters were stored in an array containing these parameters. The recorded data is temporarily stored on the memory card on the Raspberry Pi, which is then uploaded to the Google Drive storage media and then deleted from the memory card automatically to avoid the data accumulation

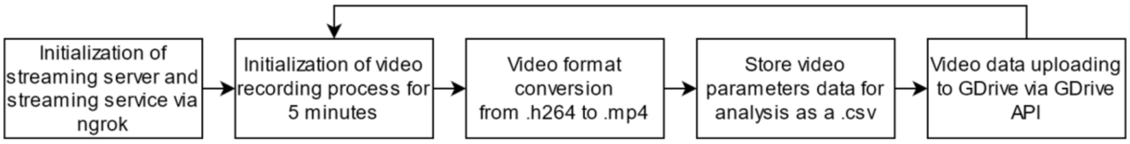
Initialization of streaming server and streaming service via ngrok
Initialization of video recording process for 5 minutes
Video format conversion from .h264 to .mp4
Store video parameters data for analysis as a .csv
Video data uploading to GDrive via GDrive API

Figure 3. The running system stages

RESULT AND DISCUSSION

The File Size Analysis

Figure 4 shows the impact of the resolution and frame rate variations on the file size produced by each scenario. The graph of the distribution of file sizes shows a relatively stable and uniform data sample for resolutions 640x480 and 800x600 in both framerates. The standard deviations for both resolutions are 0.82 and 1.28 for framerate 15 fps, and 1.13 and 1.37 for framerate 30 fps. However, the standard deviation for 1024x768 resolution significantly increases for both framerates (5.04 and 6.47, respectively). The average results of each scenario are then calculated to compare the file sizes of each tested scenario. A bar graph of the average comparison of video file sizes in megabytes can be seen in Figure 5. The video file size with a resolution of 1024x768 pixels has the biggest size for frame rates 15 fps and 30 fps.

The bar graphs in Figure 5 are the average result of 12 samples of five-minute recording data for each resolution and fps variant. The graph shows an increase in video file size by approximately 70% to 80% for each resolution variant with a different fps. At the same framerate, an increase of approximately 90% is produced at an increase in resolution of 640x480 to 800x600, both at 15 fps and 30 fps. In contrast, an increase in video file size of approximately 200 % is produced at an increase in resolution of 800x600 to 1024x768 at 15 fps. Those increasing resolution at 30 fps resulted an increase of about 245% in video size.

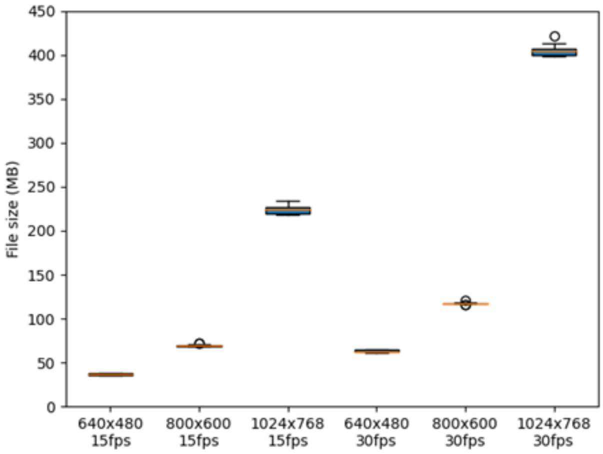

Figure 4. The file size distribution

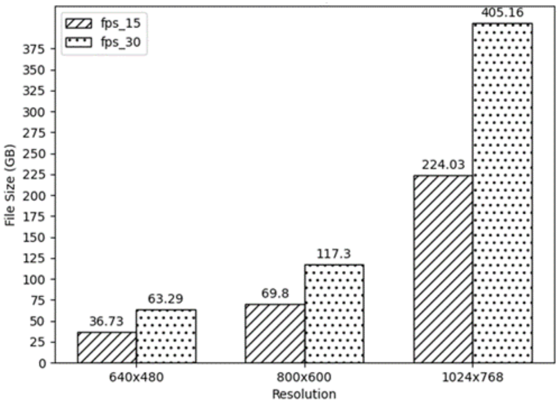

Figure 5. The average file size comparison

The Duration Analysis

Figure 6 depicts the graph of the running time distribution for each tested scenario. The total duration is the sum of the recording duration, upload duration, and conversion duration. This graph shows that the position of the median line in the box plot varies for each scenario, which means that the data are relatively unstable. In our opinion, this is caused by the instability of the internet connection when uploading video data into Google Drive storage, which can be confirmed using the graph in Figure 7. The two graphs are relatively very similar. Without including the unstable upload duration variable and the constant recording duration (300 seconds), the graph in Figure 8 shows a relatively more stable distribution than the graph in Figure 7, except for the 1024x768 30fps resolution data. The box plot of the scenario above shows there is an extreme outlier.

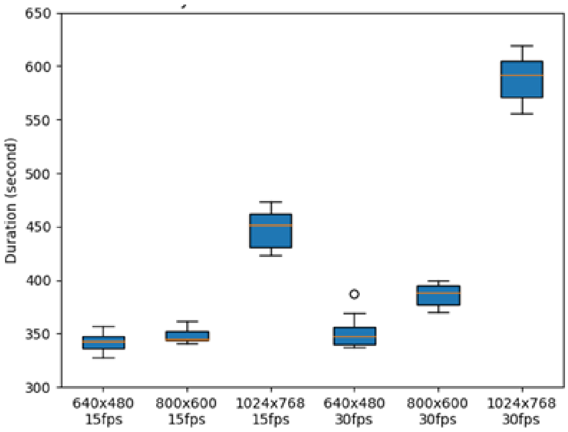

Figure 6. The system running time distribution

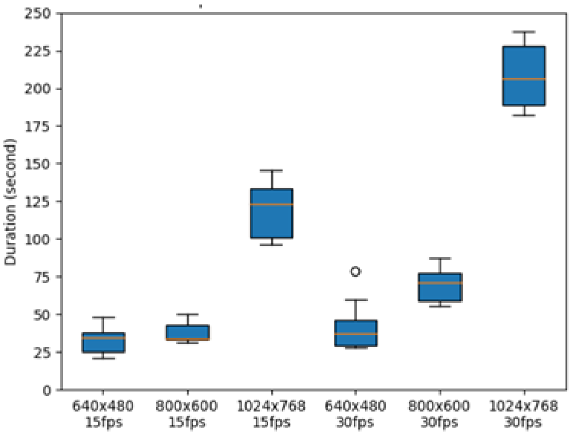

Figure 7. The uploading time distribution

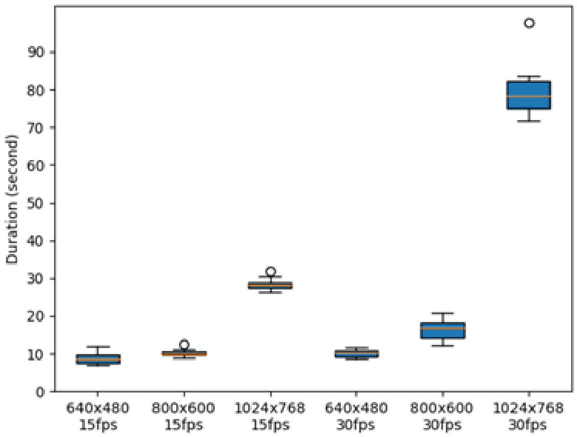

Figure 8. The system running time distribution excluded the recording and uploading duration

The Evaluation and Recommendation

Observation results show that the 640x480 and 800x600 scenarios with the two fps variants are relatively stable, with a duration of approximately 10 seconds to 20 seconds. Although the duration of the scenario of 1024x768 with 15 fps in Figure 8 does not have an extreme difference from the scenario of 640x480 and 800x600, the 1024x768 at 15 fps scenario will take a lot of space for storage memory. Figure 9 illustrates the estimated storage memory used for each scenario in gigabytes with a recording duration of 35 days based on broiler harvest age [26].

The results of the comparison of the duration of the system test run and the comparison of the file size of each scenario show that the 640x480 and 800x600 scenarios with their two fps variants are suitable choices for recording and live stream configurations of automatic remote visual monitoring systems. The 640x480 and 800x600 scenarios only differ in the file size increase of 90% for 15 fps and 85% for 30 fps, while the increase from 800x600 to 1024x768 is relatively very high at 220~244% and will take up a lot of storage memory for long-term recording.

The system test results show that the stable recording and live stream configurations are 640x480 and 800x600 resolutions. These configurations are also suitable for using the recordings as training data for the poultry movement pattern model, as stated in the research of [27]. This research suggests that small-resolution training data input can still provide high-accuracy results. In contrast, the 1024x768 configuration with each fps is relatively unstable and takes up a lot of storage memory compared to the two previous configurations. Figure 10 shows a sample of captured video of a chicken coop using the smallest resolution (640x480). The resolution and framerate configuration options can be adjusted to suit needs, the higher frame rate for more image smoothness or the higher resolution for better image quality.

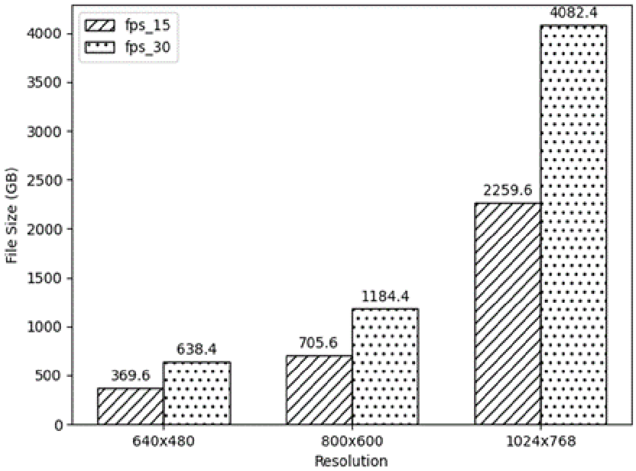

Figure 9. The file size prediction for 35 days of recording

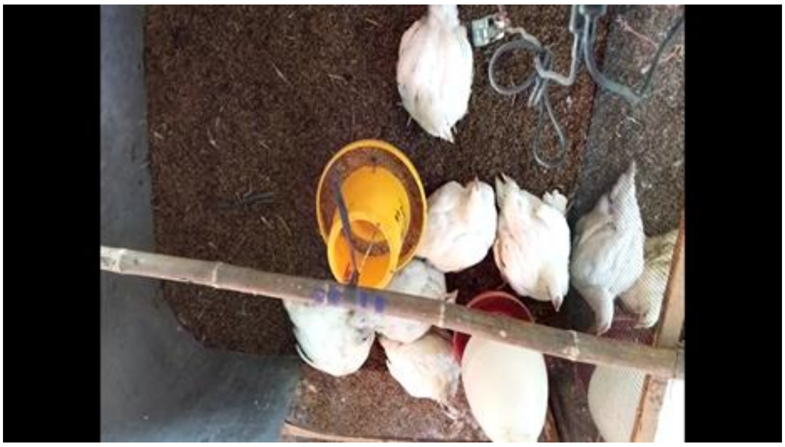

Figure 10. Captured video using the resolution of 640x480

Research on chicken farming automation and chicken coop monitoring has been conducted [11]–[13]. But none of them are based on image/video processing which is captured in a real-time fashion. Our research may be a recommended configuration for developing a remote visual monitoring system, especially for chicken cages.

CONCLUSION

The development of an automatic remote visual monitoring system using the Raspberry Pi 3B and V2 8MP Pi camera was successfully carried out. The live stream and data acquisition processes can be carried out simultaneously using the Python multithreading module. The live stream feature can be accessed publicly from anywhere using ngrok as tunneling software. The experiment's results show that stable recording and live stream configurations are 640x480 and 800x600 resolutions. In contrast, the 1024x768 configuration with each frame rate is relatively unstable and takes up a lot of storage memory compared to the two previous configurations. The Google Drive cloud infrastructure is used for storing the data acquisition results for its widespread and easy use by people with limited digital literacy, such as smallholder chicken farmers.

ACKNOWLEDGMENT

We want to express our gratitude to The Indonesian Minister of Research and Technology/National Research and Innovation Agency through IPB University for funding this research with contract number No:1/E1/KP.PTNBH/2021.

REFERENCES

[1] Directorate General of Livestock and Animal Health Statistics 2018, *Livestock and Animal Health Statistics 2018*. Ministry of Agricuture, 2018.

[2] Directorate General of Livestock and Animal Health Ministry of Agriculture of the Republic of Indonesia, *Performance Report of the Directorate General of Livestock and Animal Health 2018*. Ministry of Agriculture, 2018.

[3] J. A. Azeez and C. Akbay, "An Economic Analysis of Broiler Chicken Production for Different Production Rotations in the Northern Region of Iraq," *TEAD*, vol. 7, no. 2, pp. 76–89, 2021.

[4] M. W. Yu and F. E. Robinson, "The Application of Short-Term Feed Restriction to Broiler Chicken Production: A Review," *J. Appl. Poult. Res.*, vol. 1, no. 1, pp. 147–153, 1992.

[5] A. Ahyari, *Pengendalian Produksi*. Yogyakarta: BPFE, 1979.

[6] R. Fadilah, *Super Lengkap Beternak Ayam Broiler*. Jakarta: AgroMedia, 2013.

[7] S. Pakage, B. Hartono, Z. Fanani, B. A. Nugroho, and D. A. Iyai, "Analisis Struktur Biaya dan Pendapatan Usaha Peternakan Ayam Pedaging dengan Menggunakan Closed House System dan Open House System," *Indones. J. Anim. Sci.*, vol. 20, no. 3, p. 193, 2018.

[8] H. van Meirhaeghe, A. Schwarz, J. Dewulf, B. Venbeselaere, and M. de Gussem, *Biosecurity in Animal Production and Veterinary Medichine*. Den Haag: Acco Leuven, 2019.

[9] A. Mazick, S. Ethelberg, E. M. Nielsen, K. Molbak, and M. Lisby, "An Outbreak of Campylobacter Jejuni Associated with Consumption of Chicken, Copenhagen, 2005," *Eurosurveillance*, vol. 11,

no. 5, pp. 7–8, 2006.

[10] S. Friel and L. Ford, “Systems, Food Security and Human Health,” *Food Sec.*, vol. 7, no. 2, pp. 437–451, 2015.

[11] R. Somya, A. Ardaneswari, D. A. Saputro, and H. D. Purnomo, “Perancangan Sistem Pemantauan Pertumbuhan Ayam Pada Peternakan Ayam Broiler Dengan Pola Kemitraan,” in *Proc. Semin. Nas. Teknol. Inf. dan Multimed.* 2015, pp. 4.3-1–4.3-5.

[12] T. van Hertem, T. Norton, D. Berckmans, and E. Vranken, “Predicting Broiler Gait Scores from Activity Monitoring and Flock Data,” *Biosyst Eng*, vol. 173, pp. 93–102, 2018.

[13] J. Khairunnisa, S. Wahjuni, I. R. H. Soesanto, and W. Wulandari, “Detecting Poultry Movement for Poultry Behavioral Analysis using The Multi-Object Tracking (MOT) Algorithm,” in *2021 8th Int. Conf. Comput. Commun. Eng. (ICCCE)*, 2021, pp. 265–268.

[14] Z. H. C. Soh, M. H. Ismail, F. H. Otthaman, M. K. Safie, M. A. A. Zukri, and S. A. C. Abdullah, “Development of Automatic Chicken Feeder using Arduino Uno,” in *2017 Int. Conf. Electr. Electron. Syst. Eng. (ICEESE)*, 2017, pp. 120–124.

[15] R. Agarwala, A. Leube, and S. Wahl, “Utilizing Minicomputer Technology for Low-Cost Photorefraction: A Feasibility Study,” *Biomed. Opt. Express*, vol. 11, no. 11, p. 6108, 2020.

[16] L. C. Bishnoi, D. Singh, and S. Mishra, “Simulation of Video Transmission Over Wireless IP Network in Fedora Environment,” *IP Multimed. Commun. A Spec. Issue from IJCA*, vol. 10, pp. 9–14, 2011.

[17] N. A. Samudera, “Perancangan Sistem Keamanan Ruangan Menggunakan Raspberry Pi (bagian: Aplikasi),” *eProceedings Eng.*, vol. 2, no. 2, pp. 37–43, 2015.

[18] A. U. Bokade and V. R. Ratnaparkhe, “Video Surveillance Robot Control using Smartphone and Raspberry Pi,” in *Proceedings of 2016 Int. Conf. Commun. Signal Process. (ICCSP)*, 2016, pp. 2094–2097.

[19] M. T. Trenchanchez, A. P. Pujol, T. P. Palleja, P. M. Partinez, E. C. Clotet, and J. P. Palacin, “An Inexpensive Wireless Smart Camera System for IoT Applications Based On An ARM Cortex-M7 Microcontroller,” *J. Ubiquitous Syst. Pervasive Netw.*, vol. 11, no. 2, pp. 1–8, 2019.

[20] N. Boyko and Kryvenchuk, *Application of Cloud Services for Processing of Information Flows*, EasyChair. EasyChair, 2019.

[21] R. van Rousslet, *Pro Microsoft Teams Development: A Hands-on Guide to Building Custom Solutions for the Teams Platform*, 1st ed. APress, 2021.

[22] S. Sundararajana, U. Meyer-Baese, and G. Botella, “Custom Instruction for NIOS II Processor FFT Implementation for Image Processing,” in *Sens. Anal. Technol. Biomed. Cogn. Appl. 2016*, 2016, pp. 31–42.

[23] K. Srijakkot, I. Kanjanasurat, N. Wiriyakrieng, and C. Benjangkaprasert, “The Comparison of Faster R-CNN and Atrous Faster R-CNN in Different Distance and Light Condition,” *J. Phys. Conf. Ser.*, vol. 1475, no. 1, pp. 012–015, 2020.

[24] A. Rego, A. Canovas, J. M. Jimenez, and J. Lloret, “An Intelligent System for Video Surveillance in IoT Environments,” *IEEE Access*, vol. 6, pp. 31580–31598, 2018.

[25] K. Janard and W. Marurngsith, “Accelerating Real-time Face Detection on a Raspberry Pi Telepresence Robot,” in *Proc. 5th Int. Conf. Innov. Comput. Technol. (INTECH 2015)*, 2015, pp. 136–141.

[26] S. Nurtini, M. A. U. Muzayyanah, F. T. Haryadi, and A. Hakim, “Performance of Broiler Farmer in Partnerships System at Surakarta, Indonesia,” *J. Adv. Agric. Technol.*, vol. 4, no. 2, pp. 196–199, 2017.

[27] W. Liu, “SSD: Single Shot MultiBox Detector,” in *Computer Vision – ECCV 2016*, 2016, pp. 21–37.