Available *online* at : http://chipset.fti.unand.ac.id/

Journal on Computer Hardware, Signal Processing, Embedded System and Networking

| ISSN (Online) 2722-4422 |

Internet of Things

Automated Floor Cleaning System for Hedgehog Cages using The You Only Look Once (YOLO) Algorithm for Waste Detection

Fadlan Aziz[1], *Budi Rahmadya*[*2]

[1,2] *Computer Engineering Department, Faculty of Information Technology, Andalas University, Limau Manis, Padang, West Sumatra, 25163, Indonesia*

ARTICLE INFORMATION

Received: January 23, 2026
Revised: August 29, 2026
Available online: October 31, 2026

KEYWORDS

African pygmy hedgehog, Computer Vision, YOLOv11, WebRTC, automated cage cleaning.

CORRESPONDENCE

E-mail: budi-r@it.unand.ac.id

A B S T R A C T

African pygmy hedgehogs (*Atelerix albiventris*) have gained popularity as exotic pets; their reliance on olfactory and auditory senses makes them highly sensitive to environmental changes, particularly poor cage hygiene. Unsanitary conditions may cause odor accumulation, microbial growth, and parasitic infestations, leading to stress, reduced appetite, digestive disorders, infections, and respiratory issues. This study presents the design and evaluation of an automated cage floor cleaning system for hedgehogs based on computer vision using the You Only Look Once (YOLO) algorithm. The proposed system performs precise waste detection, autonomous cleaning, and remote monitoring and control via a mobile application. The cage consists of six PETG floor plates rotated by 180° using six MG996R servo motors, while a 12V submersible pump performs rinsing. Processing is conducted on a Raspberry Pi 4B with a PCA9685 PWM driver, integrated with Google Cloud/Firebase and a Flutter-based mobile application. Real-time video streaming is implemented using WebRTC and decoupled from the detection pipeline to maintain a stable frame rate. A YOLOv11s model was trained for object detection, achieving a precision of 0.785, recall of 0.753, and mAP@0.5 of 0.802 (mAP@0.5–0.95 of 0.491). Hedgehog detection achieved high performance (AP 0.969), while waste detection showed lower performance (AP 0.676) due to visual similarity with wooden pellets. Experimental results demonstrate smooth real-time streaming (23–27 FPS), low control latency, and reliable automated cleaning with a safety interlock mechanism that prevents cleaning when a hedgehog is detected. The system is feasible for automated cage maintenance, with future improvements focusing on dataset enhancement and system optimization.

INTRODUCTION

African pygmy hedgehogs (Atelerix albiventris) have recently become popular exotic pets due to their uniqueness [1]. However, owners often face significant challenges related to the tendency of these animals to experience stress. Stress arises as a physiological response when the cage environment does not meet their natural needs. Inadequate conditions can have a detrimental impact on the hedgehog's physical health in the long run.

One factor that can contribute to stress in hedgehogs is an unclean cage environment. Poor cage conditions, such as lack of cleanliness, can negatively affect the animal's well-being, potentially leading to health issues. Prolonged stress can result in a variety of health problems, including decreased appetite, digestive issues, and even infections.

Naturally, hedgehogs have poor eyesight but greatly rely on hearing and smell to interact with their environment [3]. Because they are very active, hedgehogs require spacious cages with natural, absorbent flooring materials, such as wood pellets. The use of sand is not recommended as it can stick to their bodies. Owners are also required to provide a clean hiding spot for isolation, especially when the animal is stressed.

Although research and technological advancements regarding exotic pets, including hedgehogs, have progressed significantly in recent years, development efforts have mainly focused on cage monitoring systems. Previous studies have developed health monitoring systems for hedgehog cages using Fuzzy Logic methods as a monitoring tool, providing real-time information via the internet. Additionally, some researchers have created prototype systems for automated waste cleaning and temperature control in rabbit cages. Temperature regulation is carried out adaptively based on cage temperature conditions using multiple sensors and Fuzzy Logic methods. Cage cleaning is performed based on the weight of the animal's waste, measured by a load cell sensor [5].

https://doi.org/10.25077/chipset.7.02.115-125.2026
Attribution-NonCommercial-ShareAlike 4.0 International. Some rights reserved

Despite the existence of automation systems, cleaning methods based on waste weight are deemed suboptimal. This is attributed to the fact that hygiene standards are not solely determined by feces, but also by urine, which significantly impacts odor pollution and humidity. Consequently, further technological development capable of more comprehensive waste detection is required to safeguard the health of the hedgehog's olfactory senses.

METHOD

The solution to the hedgehog cage cleanliness problem can be addressed through detection using Computer Vision. By utilizing Computer Vision, waste and cage bedding can be distinguished and detected more effectively. The specifications of the required equipment are presented below.

Detailed image capture of waste within the cage

To fulfill the real-time monitoring feature, a resolution of at least 720p with a frame rate of 30 fps is required. The selection of the ArduCam 16MP autofocus camera as the primary support for this project is due to its utilization of the Sony IMX519 sensor, which offers higher sharpness and autofocus capabilities that facilitate the acquisition of object images.

Table 1 Selected camera components

Spesifications	**Arducam AF 16MP** [14]
Sensor	Sony IMX519
Maximum resolution	16MP / 4656(H) x 3496(V)
Field of View (H/W)	66º(H) x 49.5 º(V)
Pixel size	1.22µm×1.22µm
Jenis focus	Autofokus (CDAF / PDAF)
Mode video	1080p@30 720p@60
IR-cut Filter	650nm

Detecting hedgehog waste with an accuracy ≥ 80%

The YOLO algorithm is a popular algorithm for single-stage detection with high accuracy and speed [19]. The captured images will be processed utilizing a Convolutional Neural Network (CNN)-based detection algorithm, specifically YOLOv7.

Table 2 Selected YOLO version

YOLO Version	**Precision**	**Recall**	**F1-Score**	**mAP50**	**mAP50-95**
YOLOv7	0,802	0,850	0,825	0,905	0,649

These performance indicators describe the following:

1. Precision: measures how effectively the model identifies correct positive results (True Positives).
2. Recall: measures how effectively the model identifies all correct positive results within a single frame; thus, a higher recall value implies fewer missed objects.
3. F1-Score: this indicator provides a final overview of the model's performance in the classification process.
4. mAP (mean Average Precision): this indicator serves as the primary metric for measuring the overall performance of the model across all existing classes.

YOLOv7 delivers balanced and accurate performance for object detection in visual imagery. Consequently, YOLOv7 is more suitable for implementation in this system, which utilizes an object dataset consisting of visual images.

Cage cleaning system automation

The cage cleaning automation system operates by inverting the floor utilizing a servo motor and rinsing it via a water pump upon the detection of waste in the absence of the hedgehog. The MG996R servo motor was selected due to its high torque capacity, which is capable of sustaining the load of two adult hedgehogs simultaneously. Furthermore, the MG996R utilizes metal gear materials, which offer superior strength and durability while maintaining a competitive price point.

Table 3 Selected servo motor components

Spesifications	**Tower Pro MG996R** [22]
Gear type	Logam (Metal)
Stall torque (4,8 V)	9.4 kg/cm
Stall torque (6 V)	11 kg/cm
Price	Rp40,000.00

For the watering system, a submersible pump with a flow rate of approximately 1325.7 Lph is utilized. The selection of this pump is further substantiated by hedgehog welfare factors; unlike diaphragm pumps which exhibit strong vibrations, the submersible pump operates with reduced noise levels. This is critical for maintaining a tranquil cage environment, thereby preventing the induction of stress in the hedgehog.

Cage control and monitoring via owner application

The designed system is capable of executing multiple simultaneous tasks utilizing a Raspberry Pi, which offers high RAM capacity. The specifications of the Raspberry Pi 4 Model B are presented below.

Table 4 Selected SBC

Spesifications	**Raspberry Pi 4B** [23]
Processor	ARMv8 Cortex-A72 1.5GHz
AI Performance	32 GFLOPS
Memory	8GB LPDDR4
Power Consumption	3-7W
Price	Rp1,550,000.00

The results of the hedgehog waste detection will be stored for notification to the owner's application. The owner's application is designed to acquire information regarding the location of detected waste within the cage, perform cage monitoring, and

https://doi.org/10.25077/chipset.7.02.115-125.2026

activate actuators; consequently, a mobile application and database are required.

Table 5 Selected application development platform

Feature	**Flutter Framework** [25]
Developer	Google
Interface (UI)	Widget
Developing speed	High speed (*Hot Reload*)
Multiplatform	Android, iOS, Web

Cage cleaning system automation

The calculation of the system's electrical power must account for the stall current, or the instantaneous peak current, occurring when all components are activated simultaneously. Based on manufacturer data, the total peak current on the 5V rail—encompassing the water pump, six MG996R servos, one SG90 servo, and the Raspberry Pi 4B—reaches 23.1 A. This value is critical for ensuring the system does not experience power failure during the initial operational transition.

The calculation of the total stall current, assuming simultaneous activation of all components, indicates that the current surge upon initial system startup can reach approximately 13.4 A. Consequently, it is concluded that the minimum power supply specification suitable for this system is 12 V 15 A, which provides a sufficient margin to accommodate current surges within the system.

General Design System

The designed system constitutes a cage capable of performing automated husk cleaning. This cleaning process is executed by utilizing images of the husk on each floor plate of the cage. Image processing is conducted using Convolutional Neural Networks (CNNs) algorithms and the YOLOv7 object detection framework. The results of the husk color change detection are transmitted to a cloud system, which subsequently generates an output in the form of a notification to the cage owner's mobile application. This notification functions to query the owner regarding the authorization of the cage cleaning operation.

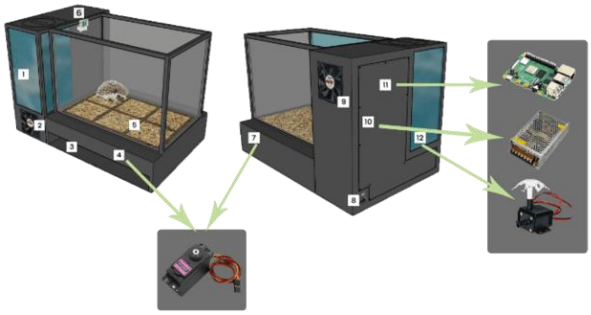


Figure 1 General hardware system design

Description:

1. Water Tank: Functions as a storage medium for flush water during the cage plate rinsing process.
2. Exhaust Fan: Functions as the cooling system for the circuit to prevent component overheating.
3. Waste Outlet Tray: Serves as a disposal repository for the residue from the cage cleaning operation.
4. Metal Gear Servo: The actuator component for the cage plates; three servo motors are located in this front section.
5. Cage Plates: Consists of six sections that will be rotated 180º upon the detection of waste on a specific plate.
6. Arducam Autofocus 16MP: Component for detecting husk color changes within the cage.
7. Metal Gear Servo: Three servo motors located in the rear section.
8. Power Button & Plug: Components to power the system.
9. Intake Fan: Functions to draw air in as a component cooling system.
10. AC to DC Power Supply: Transformer component serving as the power source.
11. Raspberry Pi 4 Model B: The main processing component within the system.
12. Water Pump: The cleaning component utilized in the cage flushing process.

Process Design

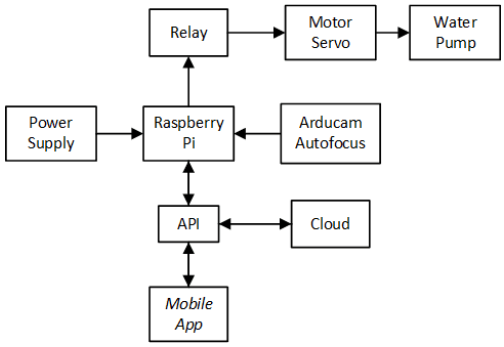


Figure 2 System block design

General System Process

The system operates by capturing images via a camera for processing by the Raspberry Pi to detect hedgehog waste, which are identified by bounding boxes corresponding to the floor plate locations. This location data is transmitted to the server via an API to issue a notification to the mobile application, thereby allowing the user to confirm the action. Upon user approval, the system executes the automated cleaning with precision based on the specific class of the floor plate detected as soiled.

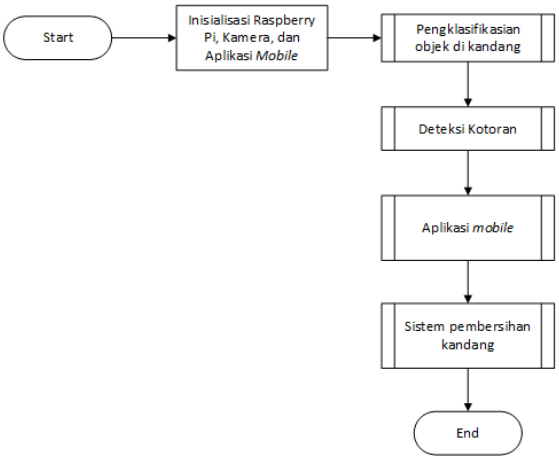


Figure 3 General system flowchart

Object Classification Process within the Cage

The development of the detection model commences with the preparation of a dataset comprising 240 hedgehog images sourced from Kaggle, specifically labeled for the hedgehog, wet husk, waste, and floor plates 1 through 6 classes utilizing bounding box annotations. The YOLO model is configured utilizing pre-trained weights and trained by adjusting parameters such as epochs and learning rate prior to evaluation via precision, recall, and F1 Score metrics. Upon validation, the model executes the inference process on new images in .jpg format to automatically detect waste; should detection occur, the system displays a bounding box accompanied by the coordinate label of the corresponding floor plate.

Waste Detection Process

This process is initiated if the image processing results on the Raspberry Pi detect the presence of waste within the cage. Bounding boxes, categorized by floor plate class, are displayed and subsequently transmitted to the cloud via an API.

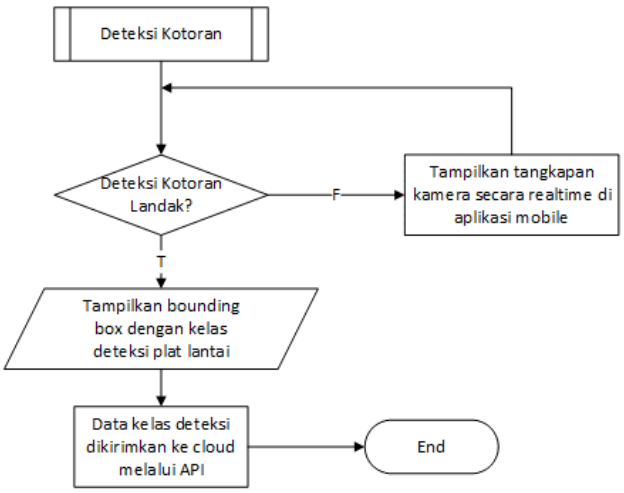


Figure 4 Waste detection flowchart

Mobile Application process

Within the mobile application, the user can observe the real-time condition of the cage. When hedgehog waste is detected by the Arducam Autofocus utilizing the YOLO algorithm, the user can view the waste bounding box based on the floor plate class via the mobile application.

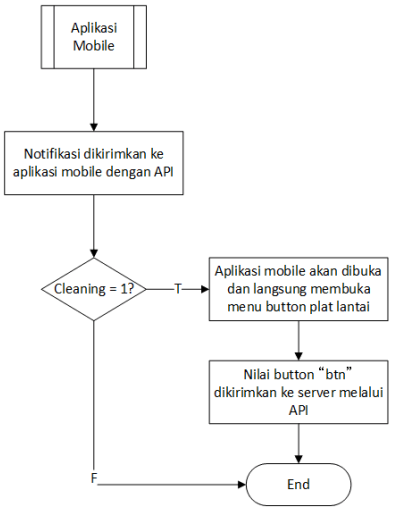


Figure 5 Mobile application flowchart

Cage Cleaning Process

The cage cleaning process initiates immediately following user confirmation via the mobile application. The value from the application button is transmitted via API to the Raspberry Pi to activate the specific servo corresponding to said input. This facilitates precise system operation, wherein only the floor plate detected as soiled is inverted and cleaned.

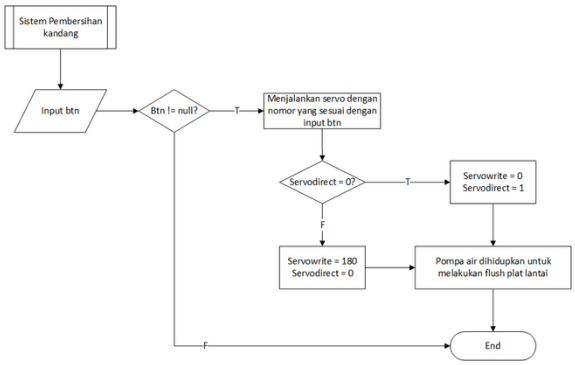


Figure 6 Cage cleaning process flowchart

Detailed System Component Design

The hardware design encompasses the physical components constituting the hedgehog cage cleaning system. This design comprises components functioning as the input and output of the system. The Raspberry Pi functions as the processing medium for data received from the input and issues output commands to the output components.

https://doi.org/10.25077/chipset.7.02.115-125.2026

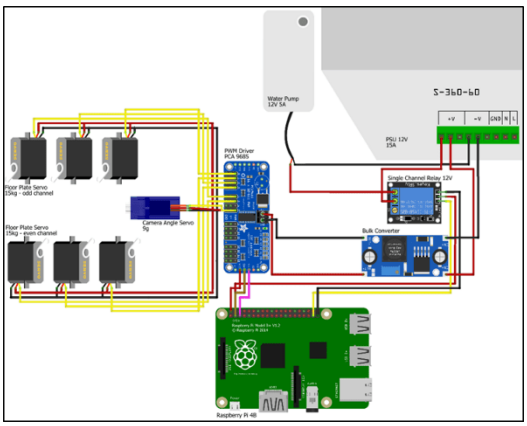


Figure 7 Hardware schematic

The following represents the main page of the application, which features two tabs: "plate pick" for selecting the floor plate to be cleaned, and "angle control" for adjusting the camera direction.



Figure 8 Main page

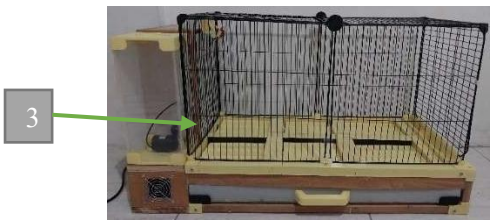
RESULT AND DISCUSSION

Hardware Implementation

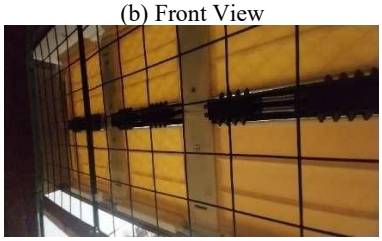
This system is implemented in a cage measuring 70 cm x 35 cm with floor plates composed of Polyethylene Terephthalate Glycol (PETG), which is water-resistant, durable, and safe for hedgehogs. The main system components include a Raspberry Pi 4B as the computing unit, a 16MP Arducam camera for waste detection, six MG996R servo units to actuate the floor plates, and a 12V water pump for cleaning. All devices are powered by a 12V 20A power supply via a buck converter and controlled utilizing a PCA9685 PWM Driver and a relay module to ensure operational stability.



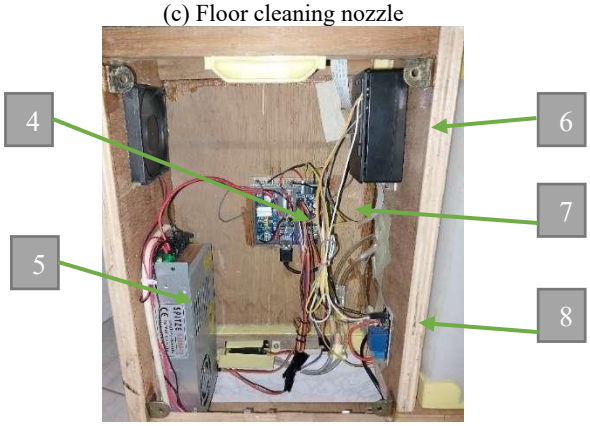
(a) Top View



(b) Front View



(c) Floor cleaning nozzle



(d) Main component management unit

Figure 9 Hardware details

Figure 9 displays the device details from various perspectives. The explanation of the utilized hardware is presented below.

1. Arducam Autofocus 16MP: Positioned at the top-left corner of the cage and connected to a camera angle adjustment servo to capture the complete image of the cage, which is processed for the video stream and waste detection.
2. MG996R Servo: Positioned at each floor plate separator to control each respective plate. There are six servo units to control the six floor plates.
3. 12V Water Pump: Located inside the water storage tank to clean the floor plate after the plate servo is inverted.
4. Buck Converter: Located inside the main component management unit, serving as a controlled voltage source for the PCA9685 and Raspberry Pi.
5. 12V 20A Power Supply: Serves as the main power source for the system, located within the component management unit.
6. Raspberry Pi 4B: Positioned in the upper section of the component management unit to prevent damage in the event of water tank leakage. The Raspberry Pi is utilized as the system's computing unit.
7. PCA9685 PWM Driver: Located alongside the buck converter. The PCA9685 is utilized to manage the seven servo motors.
8. Relay Module, positioned near the water tank line to control the water pump.

Software Implementation

1. Machine Learning Model Implementation

The object detection model was developed utilizing the YOLOv11s algorithm, selected for its superior inference speed and computational efficiency relative to preceding versions. The utilized dataset comprises 1,301 images categorized into two classes: waste and hedgehog, obtained from internal documentation and online sources.

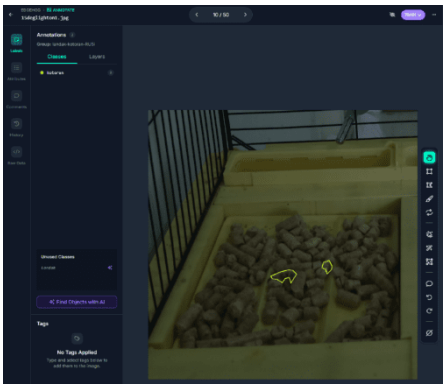


Figure 10 Dataset annotation

All images were resized to 640 × 640 pixels and annotated in YOLO format utilizing the Roboflow platform.

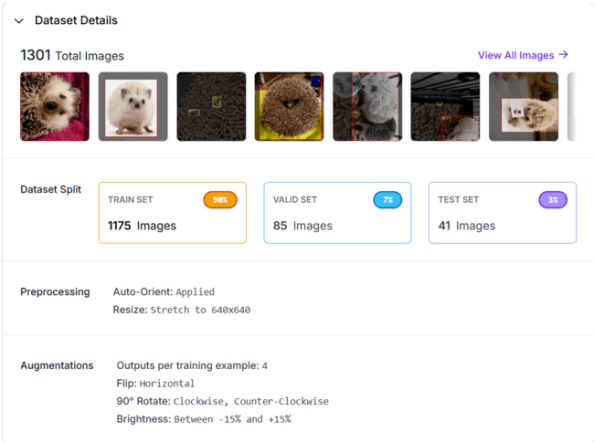


Figure 11 Train/Test Split

The preprocessing stage encompassed data augmentation, specifically size adjustments and brightness variations of ±15%. The dataset was subsequently partitioned into training (1,175 images), validation (85 images), and testing (41 images) sets. The training process was conducted on Google Colaboratory utilizing Ultralytics YOLOv11 for 10 epochs, with a total training duration of 0.064 hours.

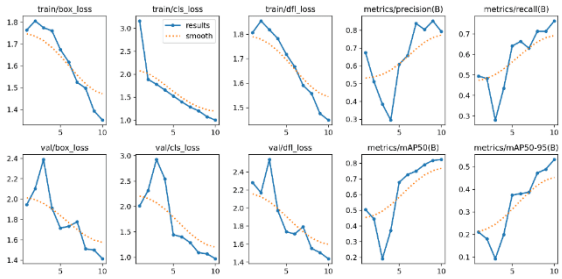


Figure 12 Custom YOLOv11 training results

Training results yielded a precision of 0.785, a recall of 0.753, and an mAP@0.5 of 0.802 (mAP@0.5–0.95 of 0.491). The best-performing model (best.pt) was selected and converted to TensorFlow Lite (TFLite) format to support inference on resource-constrained devices.

2. Implementation on Raspberry Pi

Model inference is executed on the Raspberry Pi 4B utilizing the TFLite model. To maintain visual monitoring performance, the system architecture decouples the video streaming process from AI inference. Video streaming is transmitted directly to the user application via the WebRTC protocol, whereas inference is performed periodically on high-resolution still images captured at specific camera angles.

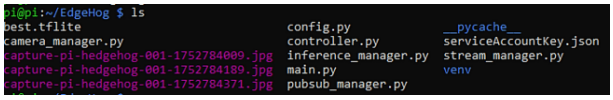


Figure 13 Script File Structure on Raspberry Pi

The software on the Raspberry Pi utilizes OpenCV for image processing, Ultralytics for model inference, Picamera2 for image acquisition, and Adafruit ServoKit for actuator control via the PCA9685. Cloud integration is achieved using Google Cloud Pub/Sub for command transmission, Firestore for detection log recording, and Cloud Storage for storing detection image evidence.

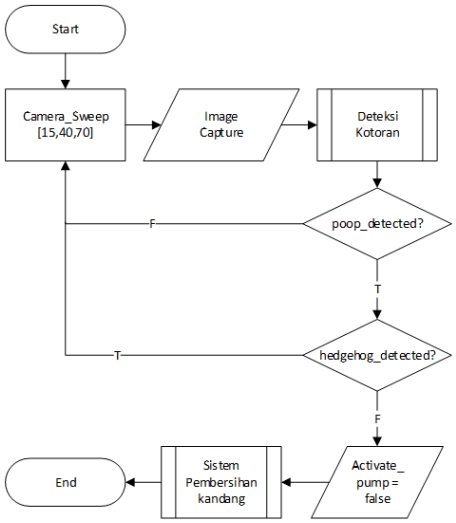


Figure 14 Automated Cleaning Implementation Flowchart

The automated cleaning system is equipped with a safety mechanism, wherein the cleaning process is executed strictly when waste is detected and no hedgehog is present within the same area.

https://doi.org/10.25077/chipset.7.02.115-125.2026

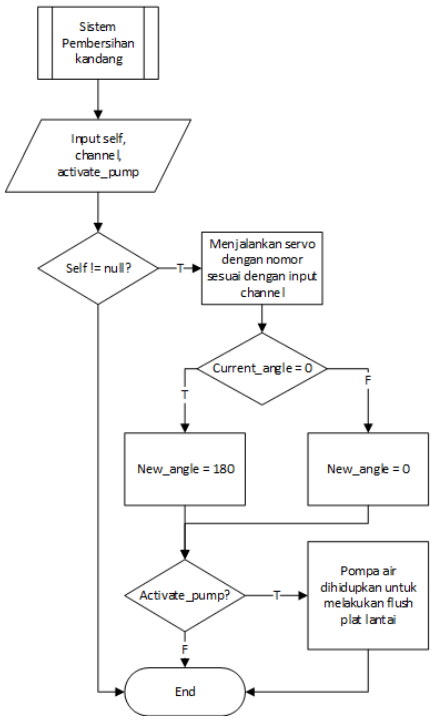


Figure 15 Modification of Cage Cleaning System Flowchart

The modification in the flowchart in Figure 15 incorporates conditional logic for water pump activation based on input derived from the preceding process. This modification ensures that the user retains the ability to initiate the floor rinsing process via the mobile application, and that automated cleaning is not constrained by water storage tank capacity.

3. Google Cloud Implementation

The system backend architecture is implemented utilizing a combination of Google Cloud Platform (GCP) and Firebase services. Google Cloud is employed to handle device-to-device communication, backend processing, and data storage, while Firebase is utilized for user management and real-time database functions.

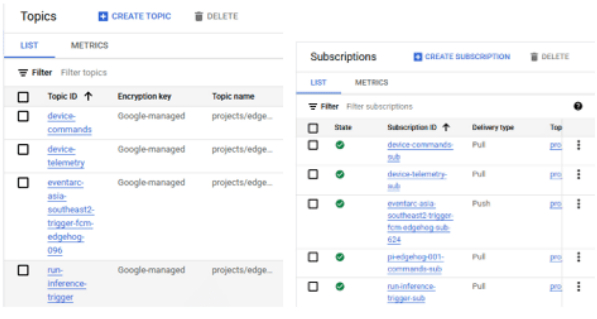


Figure 16 Managing Topics and Subscriptions on Pub/Sub

Primary communication between the mobile application and the Raspberry Pi utilizes the Google Cloud Pub/Sub service based on a publish–subscribe system. Two main topics are employed: device commands and inference triggers. The Raspberry Pi subscribes to these topics to receive commands from the application in real-time.

Backend logic is executed in a serverless manner utilizing Cloud Run. The command-handler service receives HTTP requests from the mobile application and forwards them to Pub/Sub according to the command type. Additionally, the notification-sender service is activated in an event-driven manner utilizing Eventarc to transmit notifications to the user upon the storage of new detection data in Firestore.

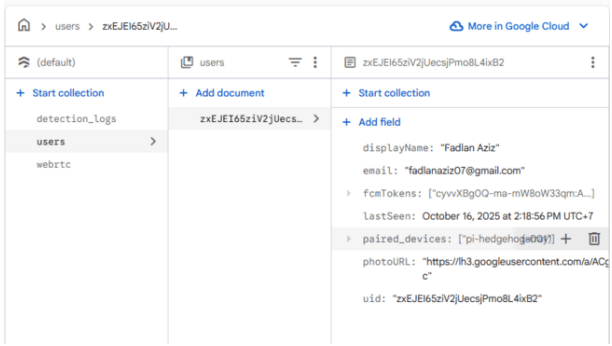


Figure 17 User Data on Firebase

Storage of detection evidence in the form of images is conducted utilizing Google Cloud Storage, while structured data such as detection logs, device status, and user profiles are stored in Cloud Firestore. Each user possesses a profile document storing a list of paired devices and notification tokens for the Firebase Cloud Messaging (FCM) service.

This cloud architecture enables the system to operate in a distributed, scalable, and secure manner, while supporting system control and monitoring from remote locations.

4. Mobile Application Implementation

The mobile application functions as the primary user interface for remote monitoring and control of the cage system. The application was developed utilizing the Flutter framework and implemented on the Android platform. User Interface (UI) and User Experience (UX) design was initially conducted using Figma to ensure an intuitive layout, navigation, and interaction flow.

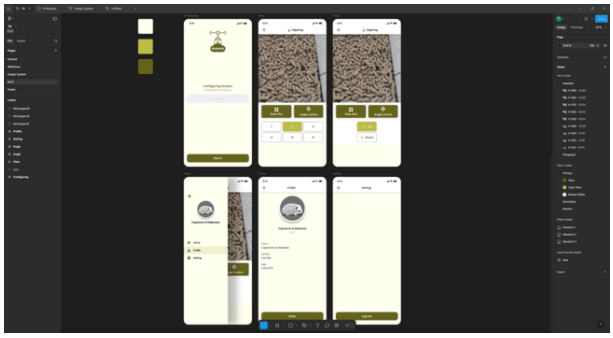


Figure 18 Initial Application Design

The application provides several key features: user authentication, device pairing, real-time video monitoring, actuator control, and detection log review. User authentication is implemented utilizing Firebase Authentication with the Google account login method to enhance security and access convenience. Upon successful login, the user is redirected to the main application page displaying the list of connected devices.



Figure 19. Mobile Application Implementation

The device pairing process is executed by entering the unique Device ID of the Raspberry Pi. This information is stored in the Firestore database and linked to the user account. This mechanism ensures that only authenticated users who have performed pairing can access device monitoring and control features.

For visual monitoring, the application receives a real-time video stream from the Raspberry Pi utilizing the WebRTC protocol. Streaming access is activated only after the pairing process is successful, ensuring connection security. As AI inference is not displayed directly on the video stream, the application provides a Detection Log page displaying the waste detection history, complete with timestamps and image evidence stored in the cloud.

The transmission of control commands, such as floor servo actuation and camera angle adjustment, is conducted via the cloud backend. Commands sent from the application are processed asynchronously to maintain user interface responsiveness.

System Testing

System testing was conducted comprehensively to evaluate software performance, inter-component communication, and system reliability under real-world operational conditions.

1. Camera Coverage Testing

Camera angle testing was performed to determine the optimal inference position for each cage floor partition. The testing was conducted by actuating the camera servo incrementally from 0° to 90° in 5° increments. At each angle, the resulting image was observed via the monitoring application to ensure that waste objects were clearly visible, in focus, and centered within the camera's field of view.

Table 6 Camera Coverage Test Results

No.	Captured Result	Camera Angle
1		5
2		10
3		15
4		35
5		40

https://doi.org/10.25077/chipset.7.02.115-125.2026

6		45
7		70
8		75

Test results indicate that angles of 15°, 40°, and 70° represent the optimal angles for the farthest, middle, and nearest areas, respectively, thereby ensuring the entire cage area can be monitored clearly.

2. YOLOv11 Model Performance Testing

This testing aims to evaluate the accuracy and reliability of the YOLOv11 model in detecting waste and hedgehog objects based on the utilized dataset. Evaluation was conducted utilizing the YOLOv11 test and validation data. The analyzed metrics encompass precision, recall, mean Average Precision (mAP), Precision–Recall Curve, and the confusion matrix. From the confusion matrix, the model performance metrics can be calculated as follows,

Class 'kotoran':

TP (True Positive) = 84

FP (False Positive) =34

FN (False Negative) = 59

$$Precision, = \frac{TP}{TP + FP}$$

$$P = \frac{84}{84 + 34} = 0.712$$

$$Recall, R = \frac{TP}{TP + FN}$$

$$R = \frac{84}{84 + 59} = 0.587$$

$$F1\ Score, F1 = \frac{2(P \times R)}{P + R}$$

$$F1 = \frac{2(0.712 \times 0.587)}{0.712 + 0.587} = 0.643$$

Class 'landak':

TP= 84; FP = 9; FN = 3

$$P = \frac{84}{84 + 9} = 0.903$$

$$R = \frac{84}{84 + 3} = 0.966$$

$$F1 = \frac{2(0.903 \times 0.966)}{0.903 + 0.966} = 0.933$$

Overall Calculation:

$$mP = \frac{0.712 + 0.903}{2} = 0.808$$

It can be concluded that the average proportion of correct positive predictions across all classes is 80.8%.

$$mR = \frac{0.587 + 0.966}{2} = 0.777$$

It can be concluded that the average proportion of positive predictions relative to the total actual positives across all classes is 77.7%.

$$mF1 = \frac{0.643 + 0.933}{2} = 0.788$$

It can be concluded that the harmonic mean value of precision and recall across all classes is 78.8%.

$$Accuracy = \frac{TP + TN}{Total} = \frac{84 + 84}{272} = 0.618$$

It can be concluded that the proportion of correct predictions relative to the total predictions is 61.8%.

The performance evaluation of the YOLOv11 model indicates an mAP@0.5 value of 0.822. The detection of the hedgehog class exhibits very high performance with an AP of 0.969, whereas the waste class possesses an AP of 0.676. The high false negative rate in the waste class is attributed to the visual similarity between waste and wood pellets, indicating a necessity for dataset quality improvement through the addition of negative data.

The TFLite model is executed utilizing a Python script on the Raspberry Pi. The camera is directed toward the center area of each floor partition to simulate operational conditions. The inference time for each detection process is recorded and analyzed.

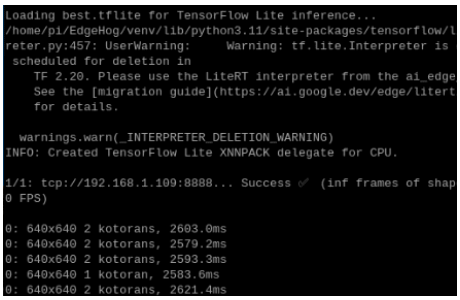


Figure 20 Model Inference Duration on Raspberry Pi

Inference testing on the Raspberry Pi 4B utilizing the TFLite model demonstrates an average inference time of 2.6 seconds per image, equivalent to 0.385 FPS. Although the FPS value is low, the strategy of decoupling streaming from inference enables the system to maintain real-time visual monitoring without interruption.

3. Application Interaction and Device Communication Quality Testing

This testing aims to evaluate the communication performance between the mobile application, cloud services, and the Raspberry Pi device, as well as to ensure all application features function according to the design. Testing was conducted by transmitting control commands from the application to the device and recording the response time. Furthermore, video streaming performance was observed via application logs to measure FPS and frame stability.

Mobile application testing indicates that the WebRTC-based video stream operates stably at a speed of 23–27 FPS with no frame loss. Control command transmission latency is recorded as low under normal conditions, approximately 61 ms; however, this may increase during network disturbances.

Detection data flow testing shows that the average time from object detection until the image evidence is successfully stored in the cloud and accessible via the user application is 7.6 seconds. Test results demonstrate that system communication operates responsively and stably. Although latency may increase due to network disturbances, the overall communication and data storage flow remains functional.

4. Pengujian Konsumsi Daya Perangkat

This testing aims to determine the system's power requirements under various operational conditions and to ensure compliance with the power supply capacity. Current and voltage measurements were conducted utilizing a digital multimeter on the 12 V power supply output under three conditions: idle, floor servo active, and water pump operational.

Table 7 Voltage and Current Measurement Results

Operational Condition	Voltage (V)	Current (A)	Power (Watt)
Idle	12	0.45	5.4
Floor servo active	12	0.82	9.8
Cleaning cycle (pump active)	12	2.45	29.4

Power consumption testing indicates a power requirement of 5.4 W in the idle condition, 9.8 W when the servo is active, and a peak load of 29.4 W when the water pump is operating. Test results demonstrate that the system's power consumption remains within safe limits and does not exceed the power supply capacity, ensuring stable system operation.

CONCLUSION

Based on the overall test results and analysis of the hedgehog cage cleaning system with waste detection utilizing the YOLO algorithm, the following conclusions are drawn:

1. The system is capable of performing automated cleaning by applying area mapping to mark floor plates where waste is detected and no hedgehog is present.
2. The system is successfully implemented between the user application and the Raspberry Pi via GCP, although latency varies depending on network conditions, with the lowest recorded latency being 61 milliseconds and the highest approximately 36 seconds.
3. The implementation of the YOLOv11 TFLite model on the Raspberry Pi 4B proves viable for the system's usage scenario. Although the average inference time is approximately 2.6 seconds, rendering it less reliable for real-time detection, this speed is sufficient for the periodic detection cycle implemented in the system.

SUGGESTIONS

Based on the testing and analysis conducted, several suggestions for future research development to improve the system are as follows:

1. The cage can be optimized further by adjusting the floor servo positioning so that the gap between floors is minimized, enabling cleaning of nearly the entire cage area.
2. Incorporate negative data (images other than the target objects present in the detection environment) into the dataset to force the model to learn patterns of ambiguous objects, such as 'waste' in this system.
3. The current system employs edge detection; it is expected that future development could involve migration to a cloud platform to alleviate the edge device workload and accelerate overall system performance.

REFERENCES

[1] G. Del Aguila, C. G. Torres, F. R. Carvallo, C. M. Gonzalez, dan F. F. Cifuentes, “Oral masses in African pygmy hedgehogs,” *Journal of Veterinary Diagnostic Investigation*, vol. 31, no. 6, hlm. 864–867, Nov 2019, doi: 10.1177/1040638719878870.

[2] G. A. Doss dan J. W. Carpenter, “African Pygmy Hedgehogs,” dalam *Ferrets, Rabbits, and Rodents*, Elsevier, 2020, hlm. 401–415. doi: 10.1016/B978-0-323-48435-0.00028-9.

[3] N. Wissink-Argilaga, “African Pygmy Hedgehogs,” dalam *Handbook of Exotic Pet Medicine*, Wiley, 2020, hlm. 13–26. doi: 10.1002/9781119389934.ch2.

[4] M. R. Muzaky, Y. A. Pranoto, dan N. Vendyansyah, “PENERAPAN IOT (INTERNET OF THINGS) PADA PEMANTAUAN KESEHATAN KANDANG HEWAN JENIS LANDAK MINI BERBASIS ARDUINO DENGAN MENGGUNAKAN METODE LOGIKA FUZZY,” *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 5, no. 2, hlm. 541–547, Okt 2021, doi: 10.36040/jati.v5i2.3732.

[5] E. D. Widianto, M. Khasanah, A. B. Prasetijo, dan R. Septiana, “Sistem Otomatisasi Pembersihan Kotoran dan Pengaturan Suhu Kandang Kelinci Berbasis Arduino Mega2560,” *Jurnal Rekayasa Elektrika*, vol. 13, no. 3, hlm. 133, Des 2017, doi: 10.17529/jre.v13i3.8422.

[6] O. I. Adebisi, I. A. Adejumobi, F. O. Durodola, A. O. Fasakin, dan O. E. Abiodun, “Development of a

https://doi.org/10.25077/chipset.7.02.115-125.2026

Microcontroller Based Automated Regulating System for Efficient Management of Poultry Operation," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 13, no. 4, hlm. 1404–1413, Agu 2023, doi: 10.18517/ijaseit.13.4.16929.

[7] M. Bhakti, "Strategi Mengendalikan Amonia di Kandang." Diakses: 8 Oktober 2023. [Daring]. Tersedia pada: https://www.medion.co.id/strategi-mengendalikan-amonia-di-kandang/

[8] M. Garbuio, B. Cordioli, E. Riva, L. Bano, dan G. Provolo, "Intra-farm dynamic of environmental conditions in commercial rabbitries with respiratory disorders using an innovative IoT device," *Smart Agricultural Technology*, vol. 13, hlm. 101725, Mar 2026, doi: 10.1016/j.atech.2025.101725.

[9] J. D. Donlon, C. G. McAloon, R. Hyde, S. Aly, B. Pardon, dan J. F. Mee, "A systematic review of the relationship between housing environmental factors and bovine respiratory disease in preweaned calves – Part 1: Ammonia, air microbial count, particulate matter and endotoxins," *The Veterinary Journal*, vol. 300–302, hlm. 106031, Okt 2023, doi: 10.1016/j.tvjl.2023.106031.

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

[11] A. R. Wasril, M. S. Ghozali, dan M. B. Mustafa, "Pembuatan Pendeteksi Obyek Dengan Metode You Only Look Once (YOLO) Untuk Automated Teller Machine (ATM)," *Majalah Ilmiah UNIKOM*, hlm. 69–75, 2019.

[12] V. A. Adibhatla, H.-C. Chih, C.-C. Hsu, J. Cheng, M. F. Abbod, dan J.-S. Shieh, "Applying deep learning to defect detection in printed circuit boards via a newest model of you-only-look-once," *Mathematical Biosciences and Engineering*, vol. 18, no. 4, hlm. 4411–4428, 2021, doi: 10.3934/mbe.2021223.

[13] J. B. ReVelle, *Quality essentials: A reference guide from A to Z*. Quality Press, 2004.

[14] Arducam Technology, "Arducam 16MP Autofocus Camera Product Brief." Diakses: 12 November 2023. [Daring]. Tersedia pada: https://docs.arducam.com/

[15] Raspberry Pi, "Raspberry Pi Camera Module 2," https://www.raspberrypi.com/products/camera-module-v2/.

[16] O. Geffen, Y. Yitzhaky, N. Barchilon, S. Druyan, dan I. Halachmi, "A machine vision system to detect and count laying hens in battery cages," *Animal*, vol. 14, no. 12, hlm. 2628–2634, 2020, doi: 10.1017/S1751731120001676.

[17] L. Cao *dkk.*, "Automated Chicken Counting in Surveillance Camera Environments Based on the Point Supervision Algorithm: LC-DenseFCN," *Agriculture*, vol. 11, no. 6, hlm. 493, Mei 2021, doi: 10.3390/agriculture11060493.

[18] W. M. Elmessery *dkk.*, "YOLO-Based Model for Automatic Detection of Broiler Pathological Phenomena through Visual and Thermal Images in Intensive Poultry Houses," *Agriculture*, vol. 13, no. 8, hlm. 1527, Jul 2023, doi: 10.3390/agriculture13081527.

[19] G. Li *dkk.*, "Real-time detection of kiwifruit flower and bud simultaneously in orchard using YOLOv4 for robotic pollination," *Comput. Electron. Agric.*, vol. 193, hlm. 106641, Feb 2022, doi: 10.1016/j.compag.2021.106641.

[20] Tower Pro, "MG90S."

[21] Futaba, "S3003 Servo."

[22] Tower Pro, "MG966R -Tower Pro." Diakses: 12 November 2023. [Daring]. Tersedia pada: https://www.towerpro.com.tw/product/mg996r/

[23] Raspberry Pi, "Raspberry Pi 4 Tech Specs." Diakses: 12 November 2023. [Daring]. Tersedia pada: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

[24] NVIDIA Developer, "Jetson Nano System-on-Module Data Sheet," Jan 2024.

[25] S. Boukhary dan E. Colmenares, "A Clean Approach to Flutter Development through the Flutter Clean Architecture Package," dalam *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, IEEE, Des 2019, hlm. 1115–1120. doi: 10.1109/CSCI49370.2019.00211.

[26] O. O. Tooki, A. A. Aderinto, I. A. Ojerinde, R. Abolade, dan D. Aborisade, "Development of an Intelligence based hexapod for Surveillance using Raspberry Pi," dalam *2024 International Conference on Science, Engineering and Business for Driving Sustainable Development Goals (SEB4SDG)*, IEEE, Apr 2024, hlm. 1–3. doi: 10.1109/SEB4SDG60871.2024.10629977.

[27] S. Kom. , M. Kom. Mira, S. Kom. , M. Kom. Chandra Gudiato, dan BCS. Brian Chan, *Computer Vision dan YOLO Menggali Potensi Computer Vision dan Implementasi YOLO untuk Pertanian Pintar*. Jawa Timur: Uwais Inspirasi Indonesia, 2023.

[28] G. I. E. Soen, M. Marlina, dan R. Renny, "Implementasi Cloud Computing dengan Google Colaboratory pada Aplikasi Pengolah Data Zoom Participants," *JITU : Journal Informatic Technology And Communication*, vol. 6, no. 1, hlm. 24–30, Jun 2022, [Daring]. Tersedia pada: https://ejournal.uby.ac.id/index.php/jitu/article/view/781

[29] Ultralytics, "Ultralytics YOLO11," https://docs.ultralytics.com/models/yolo11/.

[30] Riyadhul Fajri, Sriwinar, dan Firzha Fitria, "Pengembangan Real-TIme Object Detection System pada Perangkat Single-Board Computer," *Kajian Ilmiah Informatika dan Komputer*, vol. 4, no. 2, Okt 2023.

[31] S. Viswanadh Kandala *dkk.*, "Engineering End-to-End Remote Labs Using IoT-Based Retrofitting," *IEEE Access*, vol. 13, hlm. 1106–1132, 2024, doi: 10.1109/ACCESS.2024.3523066.