Jurnal E-Komtek
Vol. 8 No. 1 (2024) pp. 154-167
https://jurnal.politeknik-kebumen.ac.id/index.php/E-KOMTEK
p-ISSN : 2580-3719 e-ISSN : 2622-3066

Implementation of Convolutional Neural Network Algorithm for Detecting Empty Parking Area Based on Raspberry Pi

Suhartono[1], Satria Gunawan Zain[2], Ayu Futri[3]

[1,2]Department of Computer Engineering, Universitas Negeri Makassar, Indonesia, 90222

suhartono@unm.ac.id

https://doi.org/10.37339/e-komtek.v8i1.1220

Published by Politeknik Piksi Ganesha Indonesia

Artikel Info
Submitted:
08-06-2024
Revised:
26-06-2024
Accepted:
27-06-2024
Online first :
27-06-2024

Abstract

This study aims to implement a convolution neural network algorithm to detect empty parking areas based on Raspberry Pi 4 and use the Convolutional Neural Network (CNN) method of the YOLO V5 model. This research consists of several stages, starting from the potential and problem stages, needs analysis, literacy studies, building prototypes, system design, and system testing. The datasets collected were taken using smartphone cameras and webcams with a total of 645 image datasets which were divided into two categories, namely training data and validation. System testing is carried out in two conditions, namely during the day and at night. The results of the detection test for observing variations in the position of filled and unfilled vehicles obtained the highest average accuracy during daytime conditions, while for observing cars entering and leaving the parking lot during day and night conditions, the results were the same percentage of success.

Keywords: Empty Area Algorithm, Raspberry Pi 4, YoloV5, CNN Algorithm

Abstrak

Penelitian ini bertujuan untuk mengimplementasikan algoritma convolution neural network untuk mendeteksi area parkir kosong berbasis Raspberry Pi 4 dan menggunakan metode Convolutional Neural Network (CNN) model YOLO V5. Penelitian ini terdiri dari beberapa tahap, mulai dari tahap potensi dan masalah, analisis kebutuhan, kajian literasi, pembuatan prototype, perancangan sistem, dan pengujian sistem. Dataset yang dikumpulkan diambil menggunakan kamera smartphone dan webcam dengan total 645 dataset gambar yang terbagi dalam dua kategori yaitu data latih dan validasi. Pengujian sistem dilakukan pada dua kondisi yaitu pada siang hari dan malam hari. Hasil uji deteksi untuk pengamatan variasi posisi kendaraan terisi dan tidak terisi memperoleh rata-rata ketelitian tertinggi pada kondisi siang hari, sedangkan untuk pengamatan mobil masuk dan keluar tempat parkir pada kondisi siang dan malam diperoleh hasil persentase keberhasilan yang sama.

Kata-kata kunci: *Algoritma Area Kosong, Raspberry Pi 4, YoloV5, Algoritma CNN*

1. Introduction

Parking space is a requirement in open places such as retail stores, offices or workplaces, and entertainment venues where parking space must be provided by the owner of the building. The number of vehicles and the extent of the parking area can be an obstacle for drivers in finding a parking space in the unfilled part of the slot. Thus, motorists must go around to get a parking space that is not filled. Meanwhile, motorists need to find a parking space quickly and efficiently so that drivers don't have trouble causing the activity to take up less time and cause congestion in the parking lot.

Getting an empty parking area to leave the car at the vehicle's stopover is often a problem. Various studies have been conducted to determine occupied and unoccupied parking spaces, including the development of car image detection that identifies the number of occupied and unallocated parking spaces [1]. Implemented and modified the YOLO algorithm. YOLO applies a neural network to an image, then divides the image into regions and predicts the bounding boxes and probabilities of each region. Then the probability of each bounding box is calculated and classified as an object. YOLO can perform object recognition in real-time with an accuracy and speed of 45 frames per second.

Based on experiments that have been carried out using the program on 13 image datasets that demonstrate how the modified YOLO algorithm (M-Yolo) can detect the number of cars accurately, from what happens, the resulting accuracy is 100% when running the program, which displays the estimated value generated of each detected object is a four-wheeled vehicle available in the parking slot [2]. Dive the experimental results using the CPU and GPU, with an average difference of 0.179 seconds. The location of parking lots and the statistics of the number of empty and filled parking lots were not explained in this study [3].

The development of an infrared sensor-based parking availability system using Arduino for cars is the next research. Using infrared sensor modules as a detector in and out of cars from the parking lot. This study aims to create a system that can identify empty parking spaces [4]. the trumpet must be able to rotate so that the infrared module sensor can correctly identify the car based on the desired distance. This study uses an Arduino board and limits infrared sensors to detect cars entering a parking lot without using a camera. [5].

Future research is to use YOLO to automatically label parking spots on CCTV footage when it becomes available. The method developed in this study can identify parking lots based on the location of the four-wheeled vehicles in the image. By having a high accuracy value in

this study, namely an average accuracy of 93.48%. However, they still used YOLO V3 at the marking stage in this study [6].

Based on this explanation, in this study, the researchers provided a solution using the Convolutional Neural Network (CNN) method, the YOLO V5 model, which has better performance and accuracy than the previous version, namely the YOLO V3 model and uses a Raspberry Pi 4 and Webcam which makes it more accurate, practical. , and it is hoped that it can clearly display and explain occupied and unfilled parking slots, and can provide information such as the number of occupied and unfilled parking spaces (empty). Convolutional Neural Network (CNN) is a multilayer perceptron (MLP) that has been developed cooperatively and processes data in two directions. Due to its wide network depth and wide application for image data, a Convolutional Neural Network (CNN) is categorized as a Deep Neural Network (DNN) [7]. A deep learning model called a Convolutional Neural Network (CNN) is often used for image processing or visualization. The neurons in CNN are arranged in three dimensions length, width, and height. When analyzing images or images, the Convolutional Neural Network (CNN) approach is very effective and efficient [8].

The Convolutional Neural Network (CNN) algorithm works very well for object detection. In the field of computer vision, the Convolutional Neural Network (CNN) Algorithm is an algorithm that is widely used to identify objects in images [9]. By utilizing the Convolutional Neural Network Algorithm in object detection, this algorithm can perform segmentation to produce satisfactory results. Many studies have demonstrated effective object detection using the Convolutional Neural Network (CNN) algorithm. Based on the description given above, this study will use a convolutional neural network (CNN) algorithm to detect empty parking lots.

2. Method

The development process in this study consists of 6 stages, namely, literature study, preparation of tools and materials, development process, validation, testing, and finally, revision. If the revision process runs smoothly without any problems, then the process has been completed. The development procedure can be described as follows. The development procedure can be seen in **Figure 1.**

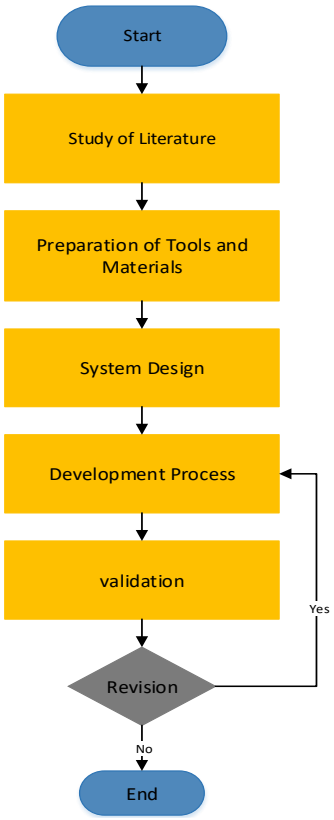

Figure 1. Research Stages Workflow

Study of literature, for this research, a literature study was conducted to obtain more information about the application of the Raspberry Pi-based convolutional neural network algorithm for detecting empty parking spaces. Preparation of Tools and Materials Then, determine what tools and materials are needed in research to develop a system that can implement the Raspberry Pi-based Convolutional Neural Network algorithm to detect empty parking spaces.

System Design and Design: In the next stage, the researcher designs the system so that scheduled interactions can be completed on time. This step involves designing the hardware, implementing the Convolutional Neural Network (CNN) algorithm to detect empty parking areas, and designing the overall interface of the prototype. There are two phases in design and engineering, namely software and hardware design.

At the design and testing stage the system that implements the neural network algorithm for empty parking detection based on Raspberry Pi is done in stages so that the results are in accordance with the developer's plan that has been proposed. The system design process that implements the convolutional neural network algorithm to detect empty parking spaces on the Raspberry Pi platform is described in the flowchart below.

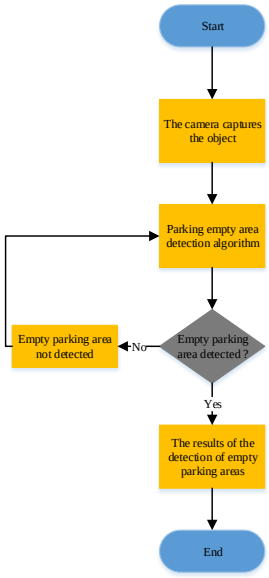

Figure 2. System Flowchart

Through **Figure 2** there is a flowchart of the overall system algorithm. The first process is that the webcam records the video of objects in real-time (directly) so that the Raspberry Pi 4 algorithm will carry out the object recognition process in the parking area. If no objects are detected in the parking area, the webcam will repeat the video recording. Conversely, if the system succeeds in detecting an empty parking area, the system selects a filled and empty space in the parking area. Then, the data obtained from the detection of empty parking areas will be stored in a txt archive format.

The system designed in this study is an implementation of the Convolutional Neural Network (CNN) algorithm for detecting empty parking lots based on Raspberry Pi. This system uses the Convolutional Neural Network (CNN) method, webcam, object recognition system YOLO V5, and Raspberry Pi. In the system to be made, the webcam is made in such a way that it can run in real time while the system is running. The system architecture is presented in **Figure 3**.

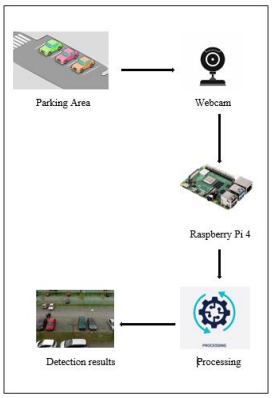

Figure 3. System Architecture

In **Figure 3**, the webcam is built to turn on in real time when the system is running. Then, a web camera installed in the parking area monitors the parking area. This is where image manipulation comes into play. Webcams collect parking lot images, which are then processed using the Convolutional Neural Network (CNN) method, which can detect empty parking lots.

The architecture of the YOLO algorithm uses a Convolutional Neural Network, which has 24 convolutional layers followed by 2 fully connected layers. The convolutional layer functions to extract features from the input image, while the fully connected layer plays a role in predicting the output probability and coordinates. The architecture of the YOLO algorithm can be seen in **Figure 4.**

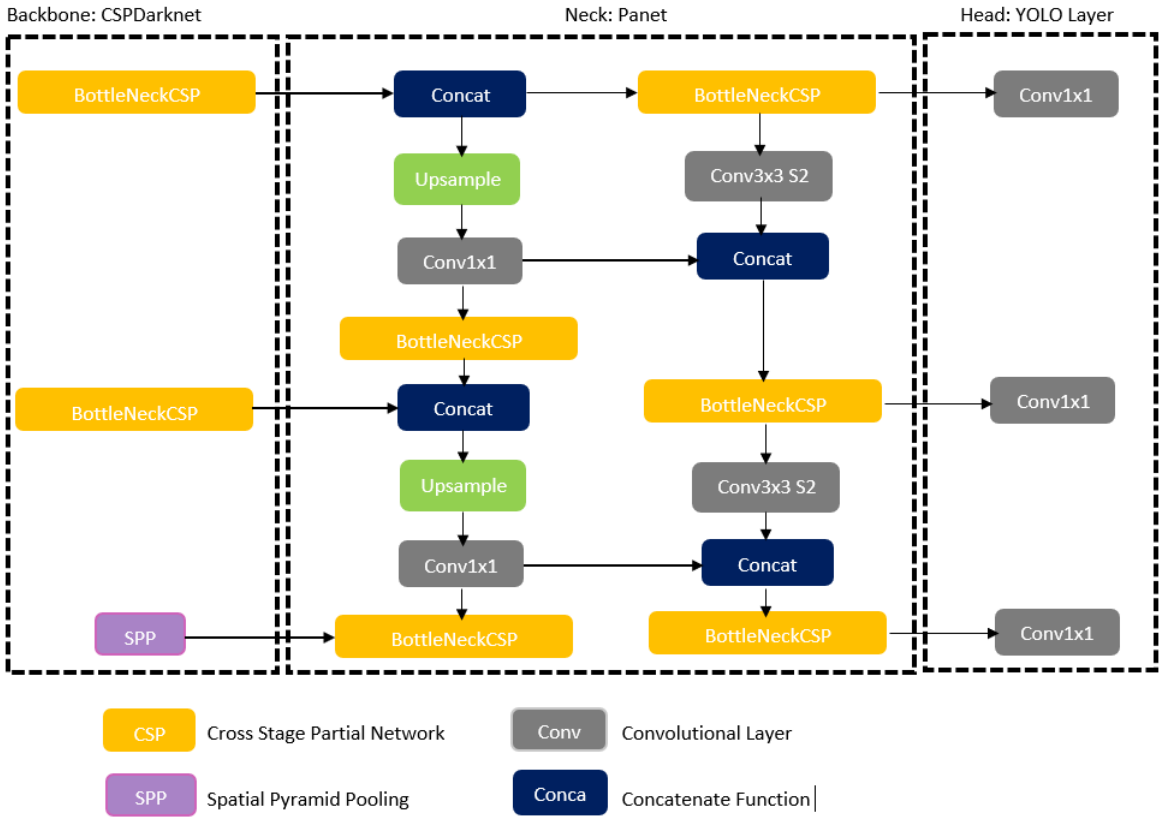

Figure 4. YOLOV5 Algorithm Architecture

Development Process, in this study, there are two development processes, namely, the hardware development process and the software development process. The system development process can be seen as follows. Hardware Design: Hardware design consists of 3 bases, namely the main system acts as input, process, and output. Software Development, the design of the software in this study has several stages in order to be able to determine the output as expected. These processes include image capture, labeling, distribution of datasets, training with CNN, testing, and prediction results.

Validation: To determine whether each component used can work properly, tool validation is carried out. Can the Raspberry Pi 4 be used in this situation to process digital images from a Logitech webcam?

Trials, System testing carried out at this trial stage is functional testing. This is to show that all the components used work very well without any problems

Revision: This stage includes repairs and adjustments if any of the tools previously described are not functioning properly and correctly, starting with the conclusion of the tool's validation. In this study, data from systematic testing was used as a data collection technique. The collected data is then presented in tabular form. The next stage is to examine the data that has been obtained after the data collection process is complete. In this study, a qualitative descriptive data analysis methodology was applied, and a table of system test results was used to collect data which was then analyzed. Analysis of image data processing performance to get recognition of empty parking areas and detection accuracy can be calculated using the formula:

$$\%Success = \left(\frac{\text{p slot Detection count is correct}}{\text{Number of parking slots}}\right) \times 100\%$$

Average Accuracy = $\frac{1}{n}\sum_{i=1}^{n} Ai$

Keterangan :

n = total number of trials

i = correct number of trials

Ai = accuracy value

3. Results and Discussion

3.1 Presenting the Results

At this stage, 645 datasets in the form of cars or four-wheeled vehicles were collected from the period August to September. The collected images are divided into 2 folders, namely 70% training data and 30% test data. From the dataset collected, the more various types and positions of four-wheeled vehicles collected, the higher the accuracy obtained during the trial or system testing process. There are three image annotation formats in image labeling software, from Pascal CreateML to YOLO). The image annotation used in this labeling is YOLO because it is more relevant to the architecture of the YOLO V5 model used in this study; the output obtained for labeling is an image annotation with a file format with the txt extension containing information from the bounding box.

The dataset collected is 645 cars or four-wheeled vehicles generally parked in campus parking areas. It doesn't stop here; the next step is the image labeling process to get the output in the form of an image annotation in a file with the extension text. Training of several models is carried out, and this aims to compare and produce the best model carried out by researchers. In this case, several stages become benchmarks in each model, namely, convolution neural network, compile, training, and result. The following modeling with 3 types of CNN architectural models can show the differences and comparisons of each model that has been trained through the graph presented in **Figure 5**.

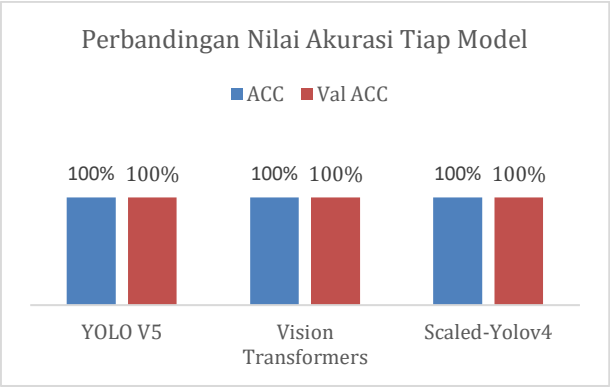

Figure 5. Train Data Comparison Chart

Four parking slots in the parking lot will be used in this study to determine how to coordinate the placement of areas marked with boxes, each of which has a different value so that they are aligned and organized in real-time. Parking slots are detected until a pop-up appears on the monitor screen while the system is running. Parking slot coordination box is presented in **Figure 6**.

Figure 6. Parking Slot Coordination Box

Figure 7 shows the results of detecting parking lots where there are four-wheeled vehicles or cars when conditions are bright (daylight). Detection of empty parking areas filled or unoccupied by four-wheeled vehicles or cars in parking slot spaces is the final step of processing

the system implementation of the Convolutional Neural Network algorithm to detect Raspberry Pi-based empty parking areas.

Figure 7. Parking Spot Detection Results

3.2 Parking Area Detection Accuracy Testing

a. Testing Based on Scenario One

The experiment was carried out five times: the first experiment was when all parking spaces were empty, the second experiment was when 3 slots were empty and 1 slot was filled, the third experiment was when 2 slots were empty, and 2 slots were filled, the fourth experimental stage was when 1 slot was empty, and 3 slots were filled, and finally, the fifth experiment occurred when all parking spaces were filled is presented in **Table 1**.

Table 1. Observation of Variations in Position When Filled and Empty Under Bright Conditions (during the day)

Testing the-	Correct Parking Slot Detection Number	Incorrect Parking Slot Detection Number	Success Percentage
1	4	-	100%
2	4	-	100%
3	4	-	100%
4	4	-	100%
5	4	-	100%
	Average Accuracy		100%

Based on the calculation above, the success rates for the first, second, third, fourth, and fifth trials are 100%, respectively.

b. Testing Based on Scenario Two

A total of five experiments were carried out, namely the first experiment when all parking spaces were empty, the second experiment when 3 slots were empty and 1 slot was filled, then the third experiment when 2 slots were empty, and 2 slots were filled, the fourth experiment

occurred when the condition of all parking lots is filled and the last fifth experiment conditions 1 slot is empty and 3 slots are filled is presented in **Table 2.**

Table 2. Observation of Variations in Position When Filled and Empty Under Bright Conditions (during the day).

Testing the-	Correct Parking Slot Detection Number	Incorrect Parking Slot Detection Number	Success Percentage
1	4	-	100%
2	4	-	100%
3	4	-	100%
4	4	-	100%
5	3	1	75%
Average Accuracy			95%

Based on the calculations above, the success rate of the first, second, third, and fifth trials is 100% each, while the fourth trial has a success rate of 75%. The system only detected three of the four four-wheeled vehicles or cars in the actual parking lot situation. This is because the shape of the object appears dark and dim due to poor light factors.

c. Testing Based on Scenario Three

Table 3 presents five experiments that were carried out when four-wheeled vehicles (cars) left the parking lot in bright conditions (during the day) and dark conditions (at night).

Table 3. Observations on Testing When the Vehicle Exits the Parking Lot in Bright (during the day) and Dark (night) Conditions.

Testing the-	Condition	Number of vehicles in the parking lot	Number of vehicles leaving the parking lot	Empty slot detection results	Filled Slot Detection Results
1	Afternoon	2	1	3	1
2	Afternoon	4	1	1	3
3	Afternoon	1	1	4	0
4	Evening	2	1	3	1
5	Evening	4	1	2	2

Based on the tests above, the success rate is seen in the difference between system detection and the actual situation. If there is a difference between the system detection and the actual situation, the data can be considered wrong. Based on the calculation results above, the success rate for the first, second, third, and fourth trials is 80%.

d. Testing Based on Scenario Four

Table 4 presents the experiment carried out when four-wheeled vehicles (cars) entered the parking lot five times in bright conditions (during the day) and dark conditions (at night).

Table 4. Observations on Testing when the Vehicle Enters the Parking Lot in Bright (during the day) and Dark (night) Conditions

Testing the-	Condition	Number of vehicles in the parking lot	Number of vehicles leaving the parking lot	Empty slot detection results	Filled Slot Detection Results
1	Afternoon	1	1	2	2
2	Afternoon	2	1	1	3
3	Afternoon	3	1	0	4
4	Evening	1	1	2	2
5	Evening	2	1	2	2

Based on the calculation results, the success rate for the first, second, third, and fourth trials is 80%. The results of the accuracy of detecting empty parking areas experience more problems in dark conditions (at night), this is due to the shape of objects that appear dark.

e. Testing Based on Scenario Five

Look for light intensity values in bright (daytime) and dark (night) conditions. To be able to find out how much the light intensity value of each parking slot is when the system successfully detects and errors or detection errors occur, and how well the system recognizes the parking area based on the value of light intensity in bright (daytime) and dark (night) conditions for each parking slots. Light intensity values in bright (during the day) and dark (night) conditions are presented in **Table 5**.

Table 5. Light Intensity Values in Bright (during the day) and Dark (night) Conditions

Sample	Light Intensity Value	Detection Results
1	80,324	True
2	116,912	True
3	119,672	True
4	132,599	True
5	117,16	True
6	114,432	True
7	105,426	True
8	148,234	True
9	90,068	True
10	20,321	False
11	107,074	True
12	20,475	False
13	31,191	True
14	45,857	True
15	56,764	True
16	14,027	False
17	32,128	True
18	20,344	False
19	26,608	True

Table 5 shows the light intensity value for each parking slot that has been cropped. Each slot value varies, with the lowest score being 14.027 and the highest being 148.234. The table above shows that values from 20.475 to 14.027 have system detection errors, while values from 148.234 to 26.608 detect no errors.

f. Testing Based on Scenario Six

The experiment was carried out four times: in the first experiment, in conditions 1 slot was empty, and 3 slots were filled; in the second experiment, in conditions 0, slots were empty, and 4 slots were filled; in the third experiment, in conditions 3 slots were empty, and 1 slot was filled; and in the fourth experimental stage, conditions 2 empty slots and 2 filled slots is presented in **Table 6.**

Table 6. Observations When Filled and Not Filled by Objects Other than Four-Wheeled Vehicle Objects in Bright (during the day) and Dark (night) Conditions

Testing the-	Condition	Sample	Correct Parking Slot Detection Number	Incorrect Parking Slot Detection Number	Success Presentation
1	Afternoon	1	3	2	75%
2	Afternoon	2	2	2	50%
3	Evening	3	3	1	75%
4	Evening	4	2	2	50%
Average Accuracy					62,5%

Based on the calculations above, the success rate in the first and third trials has a success rate of 75%, while the second and fourth trials have a success rate of 50%. The system only detects four-wheeled vehicles or cars, while other objects cannot be detected by the system. This is because the dataset used is still insufficient, so it is necessary to add other objects besides four-wheeled vehicles. Based on this, the system has an average accuracy level of 62.5% for detecting empty parking spaces in light (daytime) and dark (night) conditions presented in **Figure 9.**

Figure 9. (a) Before an Empty Parking Area is Detected, (b) After an Empty Parking Area Parking is Detected

This research produces results in the form of the application of a convolutional neural network algorithm to detect empty parking lots based on Raspberry Pi, which is used to identify empty or occupied parking lots. In this study, 4 parking slots were determined with predetermined slot zones. To identify the 4 parking slots, slots 1, 2, 3, and 4, which were previously empty or not filled with four-wheeled vehicles, are marked in green when the system is turned on but change to red when the parking slots are filled with four-wheeled vehicles or cars. When the system is running, a statement will appear that slot 1 has been filled with 4/3 available parking, which means that 3 parking slots have not been filled out of the 4 available slots in the parking lot.

In this study, the CNN model was only able to detect four-wheeled vehicles in the parking lot, while objects other than four-wheeled vehicles could not be detected. The obtained results are more accurate. The CNN model can recognize objects well if they have various variations, such as objects that differ in size, color, and position.

4. Conclusion

From the results of the research that has been done, it can be concluded that based on Raspberry Pi 4, the design results using the convolutional neural network (CNN) method to detect empty parking spaces are considered successful. The system can find every variation of parking space with the four vehicle parking spaces that have been provided. The results of testing the use of the convolutional neural network algorithm on the Raspberry Pi 4 to identify empty parking spaces with an average accuracy of 100% are obtained in bright conditions (during the day) by observing variations in filled and unfilled conditions. The average accuracy of 95% is obtained for dark conditions (night) by observing filled and unfilled conditions. Four-wheeled vehicles were tested entering the parking lot both in light (during the day) and dark (at night), and the results showed an 80% success rate. The percentage of successful testing of four-wheeled vehicles out of the parking lot in bright (during the day) and dark (night) conditions is 80%. An average accuracy of 62.5% was obtained for bright (during the day) and dark (night) conditions by observing when it is filled and not filled by objects other than four-wheeled vehicles.

References

[1] N. Afriliana, Rosalina, and R. Valeria, "Pendeteksian Ruang Kosong Parkir di dalam Ruangan," *Ultim. Comput. J. Sist. Komput.*, vol. 10, no. 1, pp. 34–40, 2018.

[2] A. Firmansyah and D. A. Pratama, "Perancangan Smart Parking System Berbasis Arduino

Uno," *J. Teknol. Pelita Bangsa*, vol. 10, no. 1, pp. 1–9, 2019.

[3] S. Jupiyandi, F. R. Saniputra, Y. Pratama, M. R. Dharmawan, and I. Cholissodin, "Pengembangan Deteksi Citra Mobil Untuk Mengetahui Jumlah Tempat Parkir Menggunakan Cuda Dan Modified Yolo," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 6, no. 4, pp. 413–419, 2019.

[4] Fathahillah, S. G. Zain, W. Setialaksana, and M. Asriadi, "Development of Internet of Things (IoT) Based Electric Equipment Control," *Int. J. Environ. Eng. Educ.*, vol. 4, no. 2, pp. 60–65, 2022.

[5] M. W. Lestari, N. D. Siahaan, and R. Sianipar, "Rancang Bangun Sistem Ketersediaan Tempat Parkir Mobil Menggunakan Sensor Infrared Berbasis Arduino," *J. Borneo Inform. Tek. Komput.*, vol. 1, no. 1, pp. 8–14, 2021.

[6] E. Tanuwijaya and C. Fatichah, "Penandaan Otomatis Tempat Parkir Menggunakan YOLO untuk Mendeteksi Ketersediaan Tempat Parkir Mobil pada Video CCTV," *J. Ris. dan Konseptual*, vol. 5, no. 1, pp. 189–198, 2020.

[7] M. Yulianti, C. Suhery, and I. Ruslianto, "Pendeteksi Tempat Parkir Mobil Kosong Menggunakan Metode CANNY," *J. Coding, Sist. Komput. Untan*, vol. 5, no. 3, pp. 48–56, 2017.

[8] E. Triansyah and Y. I. N, "Implementasi Metode Pattern Recognition Untuk Pengenalan Ucapan Huruf Hijaiyyah," *J. Ilm. Teknol. Inf. Terap.*, vol. 4, no. 1, pp. 1–10, 2017.

[9] T. N. Wenny, J. M. Parenreng, and Suhartono, "Development of Lecture Attendance System Using QR Code in Information and Computer Engineering Education Study Program of Universitas Negeri Makassar," *J. ELINVO (Electronics, Informatics, Vocat. Educ.*, vol. 7, no. 1, pp. 19–26, 2022.

[10] Suhartono, S., Zain, S. G., & Sugiawan, S.2022. Sistem Object Recognition Plat Nomor Kendaraan Untuk Sistem Parkir Bandara.*Journal of embedded system security and inteligent system.*