Journal of Integrated and Advanced Engineering (JIAE)
Vol. 3, No. 2, September 2023: 141-150
http://asasijournal.id/index.php/jiae
http://doi.org/10.5162/jiae.v3i2.108

Human Object Detection for Real-Time Camera using Mobilenet-SSD

Rachmat Muwardi[1], Joe Mada Ranseda Permana[1], Hongmin Gao[2], Mirna Yunita[3]
[1]Department of Electrical Engineering, Faculty of Engineering, Universitas Mercu Buana, Indonesia
[2]School of Integrated Circuits and Electronics, Beijing Institute of Technology, China
[3]School of Computer Science and Technology, Beijing Institute of Technology, China

Abstract
Technology development is very rapid, so all fields are required to develop technology to increase the effectiveness and efficiency of work. One of the focuses is related to image processing technology. We can benefit from this system, so various fields have implemented image processing systems, such as security, health, and education. One of the current obstacles is safety, namely in searching for people, which is still done manually. Searching for teams to find people is often challenging because of the significant search area, low light conditions, and complex search fields. Therefore, we need a tool capable of detecting humans to assist in finding people. Therefore, to detect human objects, the authors try to research human object detection using a simple device for the human object detection system. The authors use the MobilenetV2-SSD, where this algorithm has high detection and accuracy. Using the mobilenetV2-SSD simulation method for human object recognition, a detection rate of 100% is obtained with an FPS value of 5.

Keywords:
Computer Vision;
Image Processing;
Mobilenet;
Object Detection;
Optimization;
SSD;

Article History:
Received: August 5, 2023
Revised: September 9, 2023
Accepted: September 10, 2023
Published: September 15, 2023

Corresponding Author:
Rachmat Muwardi
Electrical Engineering Department, Universitas Mercu Buana, Indonesia
Email:
rachmat.muwardi@mercubuana.ac.id

INTRODUCTION

In this digital era, technology is increasingly sophisticated and diverse. Technology can make human work more accessible. The development of object detection technology is currently experiencing a significant increase. Object detection is a computer vision technique for finding instances of objects in images or videos. Object detection algorithms usually use machine learning or deep learning to produce meaningful results. The goal of object detection is to replicate the intelligence that humans have in seeing objects using a computer. Object detection works because object detection locates an object's presence in the image and draws a bounding box around that object. Image classification and object detection scenarios look similar. In general, classification is classifying images into specific categories. Object detection technology has been used in various fields to improve the provision of services in multiple places. Object detection is one of the fundamental problems of computer vision. Some of these include object detection applications such as pedestrian detection, people counter, face detection, writing detection, pose detection, or license plate recognition [1, 2, 3, 4].

A digital image is a matrix in which the row and column indices represent a point in the picture, and the matrix elements (referred to as image elements or pixels) represent the gray level at that point. For a digital image, each pixel has an integer value, namely the gray level, which indicates the amplitude or intensity of the pixel. The idea is a twodimensional function in which the two variables, namely the amplitude value and the coordinates, are integer values [5, 6, 7].

This project will design and implement a human detection system. The system will be created to capture image irritants, find people's positions and the number of people in a room. There are various object detection algorithms, such as YOLO, R-CNN, Mask R-CNN, MobileNet, and SqueezeDet [8, 9, 10, 11, 12]. This project will use the SSD (Single Shot Detector) method. SSD is a popular detector that can predict several classes. This method uses a single deep neural network to detect objects in an image by discriminating the output space of the bounding box into a series of standard frames with different aspect ratios and scales for each location on the feature map [13].

The object detector generates a score for the presence of each object category in each standard field and adjusts the area to match the object's shape. The network also combines predictions from multiple feature maps at different resolutions to process things of various sizes. SSD detectors are easy to train and can be integrated with software systems that require an object detection component. SSD is much more accurate than other one-step methods, even with smaller input image sizes [14][15].

This research will create a device that is used to detect objects. The method used is SSD (Single Shot Detector), which can run a convolution network on the input image only once and calculate the feature map. SSD is the better choice as we can run it on video, and the trade-off fidelity is very simple. This research requires a camera, an ARM device (NanoPi M4V2), and a screen (monitor) that can display object detection results, accuracy values, and fps (frames per second) values from the camera.

METHOD

SSD (Single Shot Detector)

The SSD approach is based on a feed-forward convolution network that generates a fixed-sized bounding box set and scores for the existence of instances of an object class in those boxes, followed by a non-maximum suppression step to produce the final detection [16]. The initial network layer is based on the standard architecture used for high-quality image classification (cut before any classification layer), which we will refer to as the base network. Then add additional structures to the network to produce detection in Figure 1 with the following main features:

- Detector – The network is a detector that also classifies detected objects.
- Single-shot detectors are faster and more accurate.
- SSD predicts category scores and box offsets for a fixed number of default bounding boxes using convolution filters applied to the feature map.
-

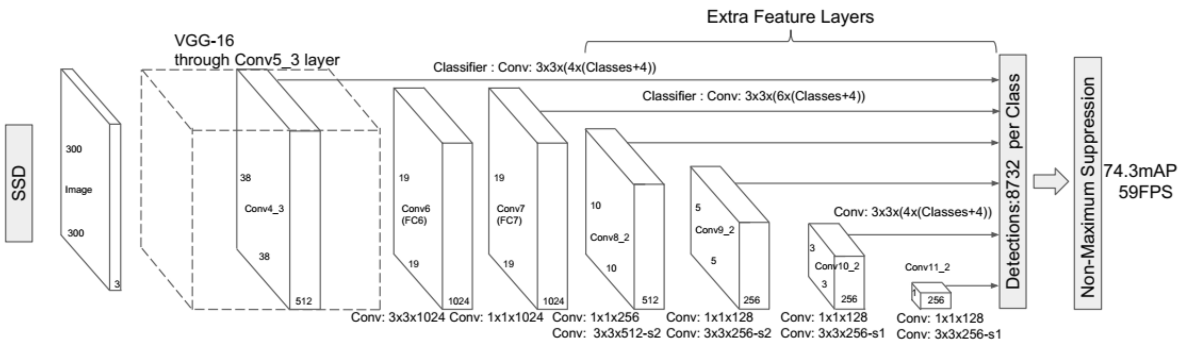


Figure 1. Architecture SSD

- We generate predictions of different scales from feature maps of various dimensions to achieve high accuracy and then separate the predictions by aspect ratio.
- These features result in high accuracy, even in low-resolution input images.

MobileNetV2

MobileNet is a convolutional neural network (CNN) architecture that can address excessive computational resource requirements. Researchers at Google have created a CNN architecture that can be used on mobile phones [17]—mobile net as a network model. Mobilenet will try to extract features that will be classified later.

MobileNetV2 uses depth wise and pointwise convolution. MobileNetV2 adds two new features, namely:

- linear bottlenecks
- shortcut connections between bottlenecks

In Figure 2, the bottleneck section, there are inputs and outputs between models, and the inner layer or layers encapsulate the model's ability to change information from lower-level concepts (pixels) to higher-level descriptive.

Open CV

Signal processing with input in the form of an image and transformed into another image as output with specific techniques. Digital image processing is carried out to correct image signal data errors that occur due to transmission and during signal acquisition, as well as to improve the quality of image appearance so that it is easier for the human visual system to interpret by manipulating and analyzing images.

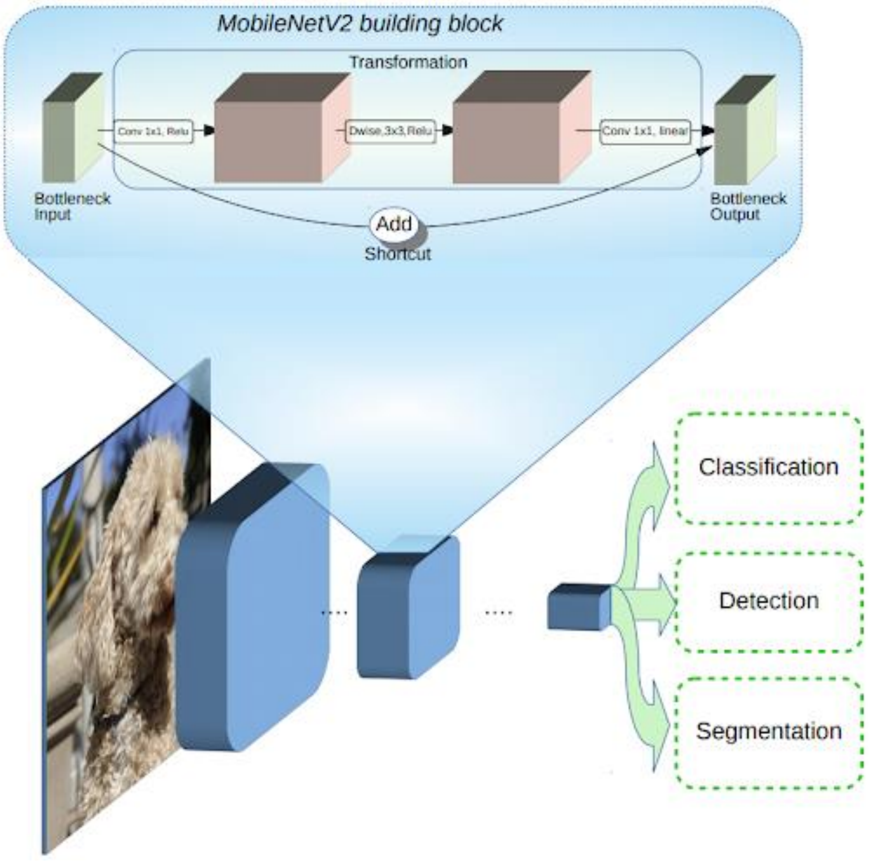


Figure 2. Architecture MobilenetV2

Operations performed to transform an image into another embodiment can be grouped based on the purpose of the transformation or the scope of operations served on the image [18].

Image Processing

A form of processing or signal processing with input in the form of an image (image) and transformed into another image as output with a specific technique. Digital image processing is carried out to correct image signal data errors that occur due to transmission and during signal acquisition, as well as to improve the quality of image appearance so that it is easier for the human visual system to interpret by manipulating and analyzing images. Operations performed to transform an image into another image can be grouped based on the purpose of the transformation or the scope of operations served on the image [19][20].

Hardware Design and Analyst

Firstly, the program must import the library, then load the necessary dataset. We can identify whether the code is valid by running it in Python. Then we put the loop “While true” to avoid exiting by itself. Inside the loop, we put the core functionality. It is as follows: (1) capture video by frame, (2) detect multi-scale, (3) draw a rectangle around a human, (4) display the resulting frame.

The device is connected to the network, USB Camera (C922), and SSD through a GPIO pin. Then for, the native display, it relates to the LG monitor through HDMI port. The program design is as follows in Figure 3 and Figure 4.

Input: in this section, there is an input device in the form of a camera that takes pictures in real-time. Aside from that, a video is also entered into the process section to be analyzed. Lastly, in the form of a datasheet, the data sheet section contains the calculation results from the object detection being investigated and used as the Real score of the video, which will later be compared with the reading results from the process section.

Process: in this section, use a laptop containing software to process Python, OpenCV, and YoloV4-Tiny programs. The program analyzes input in video or real-time from the camera. Output: In this section, a monitor displays the object detection results with an algorithm embedded in the laptop to process object detection.

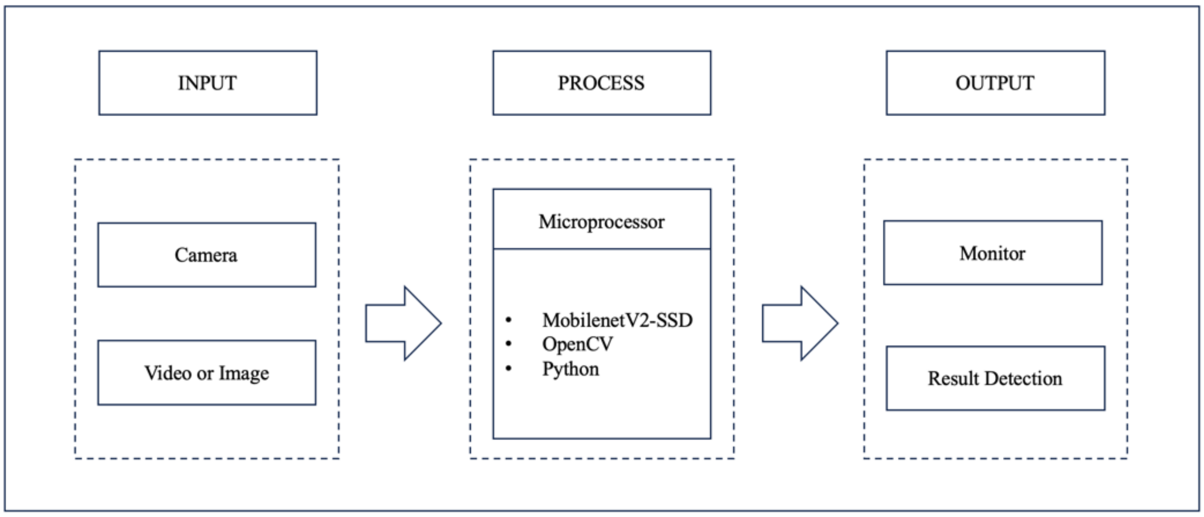


Figure 3. Diagram Block

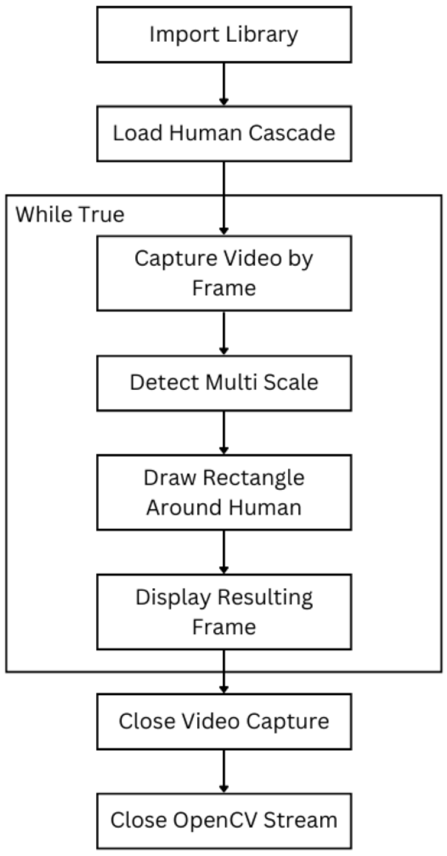


Figure 4. Program Architecture

The reading results are shown as an image on the monitor and data on the datasheet. The datasheet output contains the score obtained from the processing results in the form of Recall, Precision, Accurate, the number of detected objects, and the FPS of the processed video.

Initial setting handler. This function is necessary for creating the initial setting or applying the saved set. This function will also handle system calls for the framework's initial start. Optionally, this will also contain the necessary process for analysis as a time recorder: processor core and hardware identification. The framework needs to identify the specification of the processor. Then this function will create the hardware configuration file the framework will utilize.

The process will also prepare the other entity's necessary system call and hardware utilization function—image retrieval by a camera. Image data will be retrieved by this function inside the framework. This function will analyze and settle the hardware communication and system call necessary for the framework. Core utilization analysis. This function will monitor the processor’s utilization data and determine the available core. We can set the policy for each core’s utilization. For example, the core with high utilization will not be utilized for the task. This function also provides data for the workload assigner function. Workload division mapping and assigner for each core. This function will receive data from the core utilization analysis function. This will determine which task that will be executed by which core. This function will manage the assignment on the OS layer and process utilization, so it can pinpoint to which core the given task will run. Execute image processing task. The image processing function will run at a given core.

Retrieve and arrange the raw result and data of the task. Because of the parallelization, the development of each core's process will be retrieved, and those will be combined into necessary data for further processing. Further result processing. This function will handle the result finalization and data shaping as intended. This function will check the finalized data and running time of one cycle of the framework. This function will also handle functions necessary for analysis purposes.

RESULTS AND DISCUSSION

In the process of testing, this accuracy level is used to determine the level of accuracy of the device that has been made so that it can be said that the device can be appropriately used with a perfect level of accuracy. From the existing calculations will be obtained the results: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). If these are obtained, the existing Recall (R) and Precision (P) can be considered to determine an algorithm's accuracy level. The definition is as follows

$$R = TP/(TP + FN) \tag{1}$$

$$P = TP/(TP + FP) \tag{2}$$

Mean Average Precision (mAP) is used to evaluate the model. The purpose of mAP is as follows:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{3}$$

Figure 5 and Figure 6 discusses the video that was run for 6 seconds with 30 FPS (Total of 180 FPS). It can be detected a human image object using a video. This can be done using a real-time camera without reducing the accuracy level.

The MobilenetV2-SSD algorithm obtains satisfactory results because it obtains improvisation from the big core and little core so that the processing is more focused and improves the processor performance much better to beat YOLOV4.

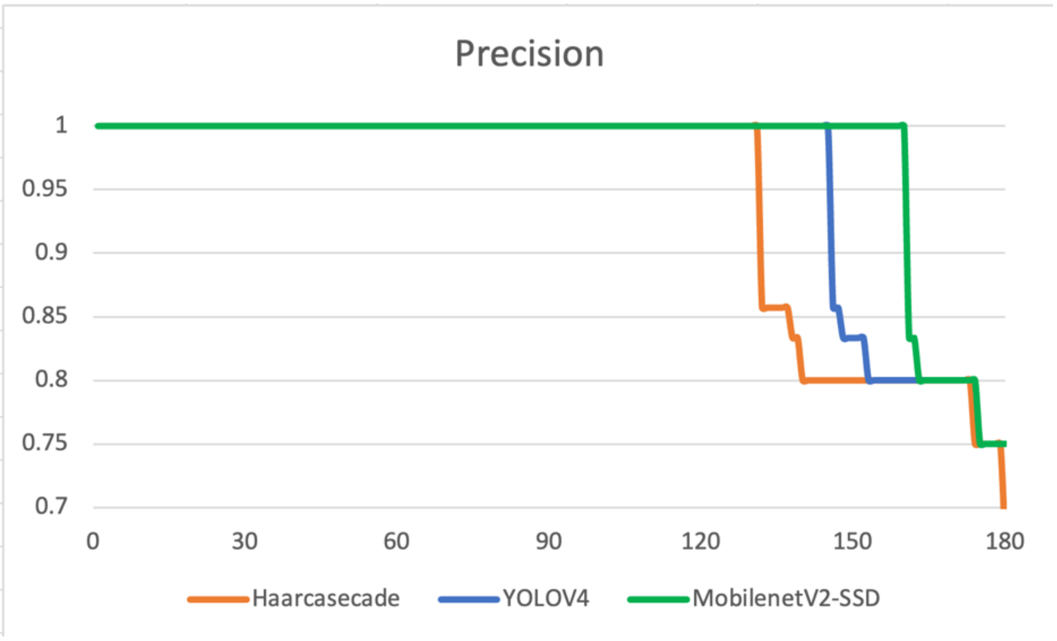


Figure 5. The precision of several algorithms

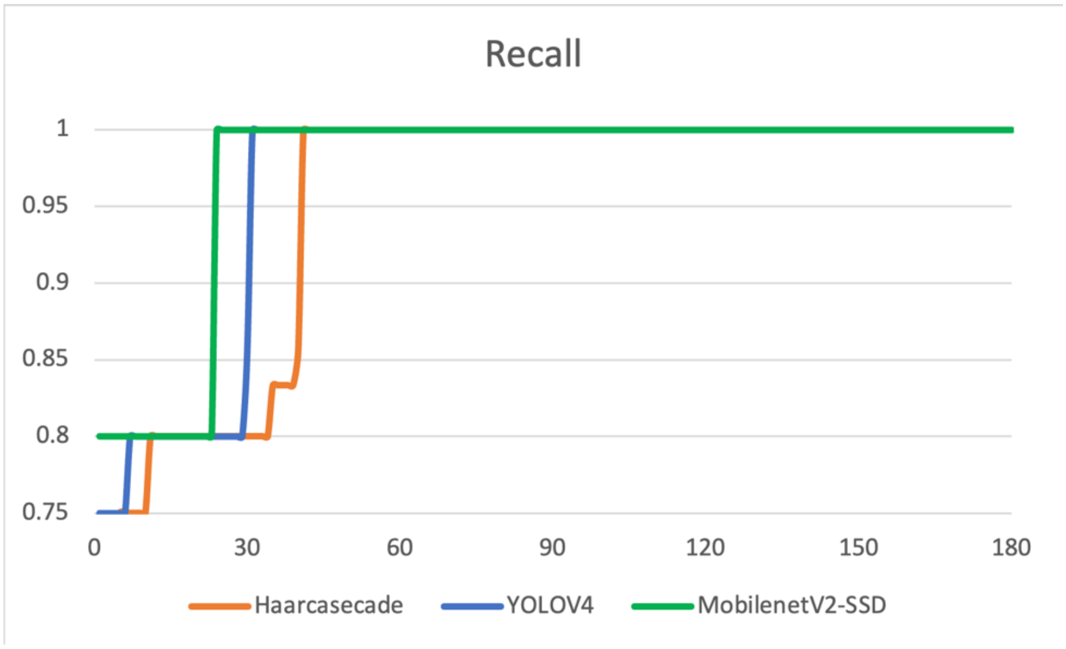


Figure 6. Recall several algorithms

Figure 7 explains the evaluation of the human object model, which is run with several algorithms that are used. It can be seen that MobilenetV2-SSD is still superior to the others because there is processor improvisation and different architecture algorithm, so the image process is much better and more remarkable compared to the YOLOV4 version, which is superior to MobilenetV2-SSD.

When the process image is in Figure 8 then the video will be processed by the program, and the processing results will be displayed in the form of the video's value, the accuracy level, and the detected object's name.

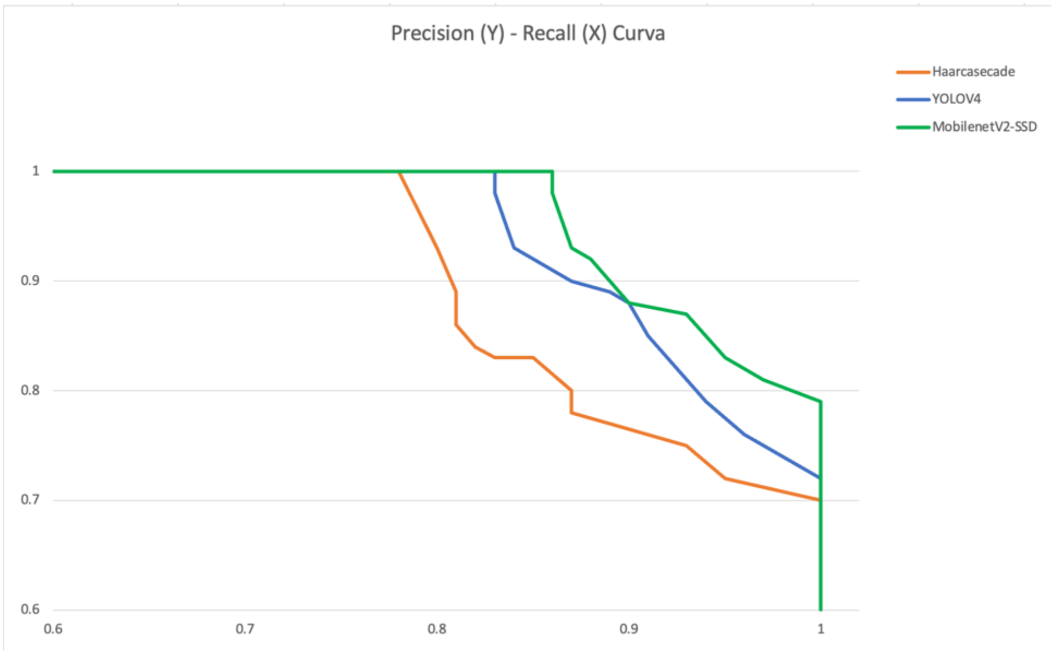


Figure 7. PR Curve of several Algorithms

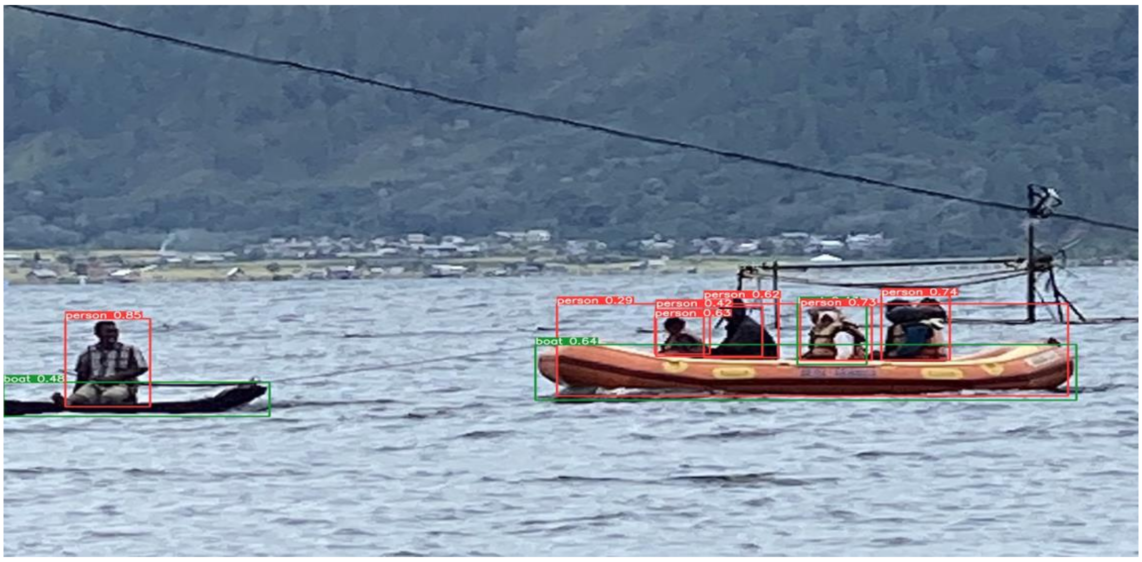


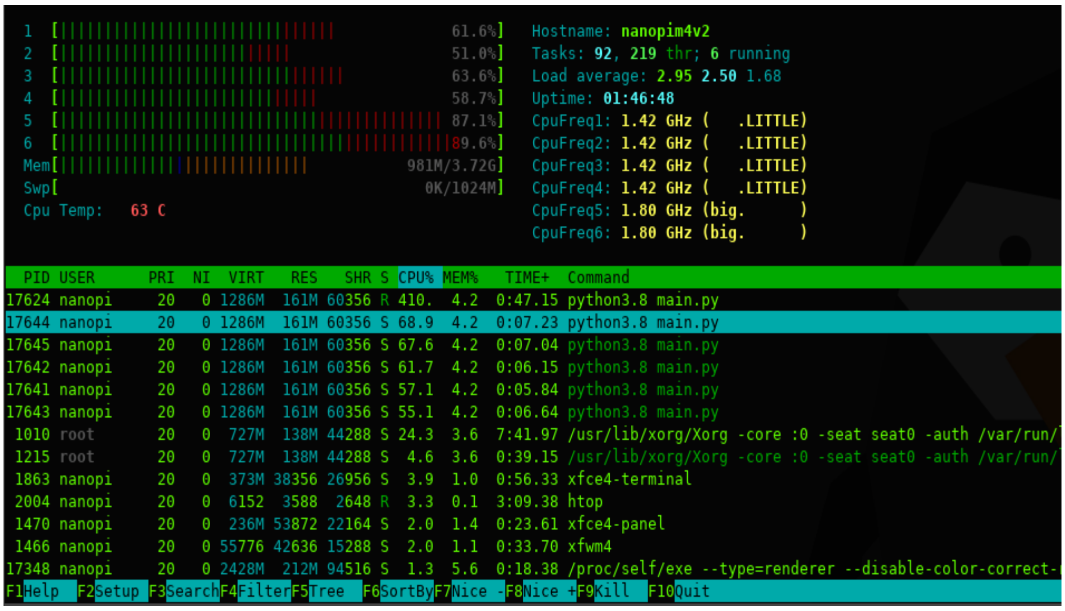
Figure 8. Human Detection

Table 1 describes the mAP of an algorithm used and the average time needed to read images with a total of 180 frames (30 FPS) contained in a video with a duration of 6 seconds. The author will discuss the performance of RAM (Random Access Memory) and CPU (Central Processor Unit) when the digital image processing is executed in Figure 9.

Explaining the image data obtained, the CPU usage on the device is not too high or not up to 100%, and the RAM that is used is only 25% of 100% or 1,02Gb of the available 4Gb RAM aside from that, the temperature on the device also does not overheat. So overall, the program and device run well, and there are no problems with their usage.

Table 1. Evaluation Result

Model	Haar cascade	YOLOV4	MobilenetV2-SSD
mAP	0.9281	0.9619	0.9705
Average Time	40 ms/frame	54 ms/frame	67 ms/frame


```
  1  [|||||||||||||||||||||||||||||||          61.6%]   Hostname: nanopim4v2
  2  [||||||||||||||||||||||||||               51.0%]   Tasks: 92, 219 thr; 6 running
  3  [||||||||||||||||||||||||||||||||         63.6%]   Load average: 2.95 2.50 1.68
  4  [|||||||||||||||||||||||||||||            58.7%]   Uptime: 01:46:48
  5  [|||||||||||||||||||||||||||||||||||||||||||| 87.1%]   CpuFreq1: 1.42 GHz (   .LITTLE)
  6  [||||||||||||||||||||||||||||||||||||||||||||89.6%]   CpuFreq2: 1.42 GHz (   .LITTLE)
  Mem[|||||||||||||||||||||||||||           981M/3.72G]   CpuFreq3: 1.42 GHz (   .LITTLE)
  Swp[                                         0K/1024M]   CpuFreq4: 1.42 GHz (   .LITTLE)
  Cpu Temp:    63 C                                       CpuFreq5: 1.80 GHz (big.      )
                                                          CpuFreq6: 1.80 GHz (big.      )

  PID USER       PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
17624 nanopi      20   0 1286M  161M 60356 R 410.  4.2  0:47.15 python3.8 main.py
17644 nanopi      20   0 1286M  161M 60356 S 68.9  4.2  0:07.23 python3.8 main.py
17645 nanopi      20   0 1286M  161M 60356 S 67.6  4.2  0:07.04 python3.8 main.py
17642 nanopi      20   0 1286M  161M 60356 S 61.7  4.2  0:06.15 python3.8 main.py
17641 nanopi      20   0 1286M  161M 60356 S 57.1  4.2  0:05.84 python3.8 main.py
17643 nanopi      20   0 1286M  161M 60356 S 55.1  4.2  0:06.64 python3.8 main.py
 1010 root        20   0  727M  138M 44288 S 24.3  3.6  7:41.97 /usr/lib/xorg/Xorg -core :0 -seat seat0 -auth /var/run/
 1215 root        20   0  727M  138M 44288 S  4.6  3.6  0:39.15 /usr/lib/xorg/Xorg -core :0 -seat seat0 -auth /var/run/
 1863 nanopi      20   0  373M 38356 26956 S  3.9  1.0  0:56.33 xfce4-terminal
 2004 nanopi      20   0  6152  3588  2648 R  3.3  0.1  3:09.38 htop
 1470 nanopi      20   0  236M 53872 22164 S  2.0  1.4  0:23.61 xfce4-panel
 1466 nanopi      20   0 55776 42636 15288 S  2.0  1.1  0:33.70 xfwm4
17348 nanopi      20   0 2428M  212M 94516 S  1.3  5.6  0:18.38 /proc/self/exe --type=renderer --disable-color-correct-
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```
Figure 9. MobilenetV2-SSD Performance

CONCLUSION

The experiments' results show that the device and system that have been made can work well to detect human objects. The NanoPi M4V2 device has an FPS (Frame Per Second) rate reasonable for digital image processing in video format with the MobilenetV2-SSD algorithm. The accuracy level in the algorithm using the NanoPi M4V2 device has a high level of accuracy with video input with brighter lighting. When the device is run, there is no problem with the CPU usage on the NanoPi M4V2 using the MobilenetV2-SSD algorithm to detect human objects.

ACKNOWLEDGMENT

The first special thanks go to Universitas Mercu Buana for supporting foreign collaborative research, the second to the Beijing Institute of Technology, Mirna Yunita, and Homngmin Gao for their help and cooperation during this research. Hopefully, there will always be a paper collaboration with the Beijing Institute of Technology in further study.

REFERENCES

[1] R. Muwardi, M. Yunita, H. U. Ghifarsyam and H. Juliyanto, "Optimize Image Processing Algorithm on ARM Cortex-A72 and A53," *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)*, vol. 8, no. 3, pp. 399-409, 2022, doi: 10.26555/jiteki.v8i3.24457

[2] R. Muwardi, H. Qin, H. Gao, H. U. Ghifarsyam, M. H. I. Hajar and M. Yunita, "Research and Design of Fast Special Human Face Recognition System," *2020 2nd International Conference on Broadband Communications, Wireless Sensors and Powering (BCWSP)*, Yogyakarta, Indonesia, 2020, pp. 68-73, doi: 10.1109/BCWSP50066.2020.9249452.

[3] R. Muwardi, H. Gao, H. U. Ghifarsyam, M. Yunita, A. Arrizki and J. Andika, "Network Security Monitoring System Via Notification Alert," *Journal of Integrated and Advanced Engineering (JIAE)*, vol. 1, no. 2, pp. 113-122, 2021, doi: 10.51662/jiae.v1i2.22

[4] F. A. Bohani, S. R. Yahya and S. N. H. S. Abdullah, “Microgrid Communication and Security: State-Of-The-Art and Future Directions,” *Journal of Integrated and Advanced Engineering (JIAE)*, vol. 1, no. 1, pp. 37-52, 2021, doi: 10.51662/jiae.v1i1.11

[5] S. Suganyadevi, V. Seethalakshmi and K. Balasamy, "A review on deep learning in medical image analysis," *International Journal of Multimedia Information Retrieval*, vol. 11, no. 1, pp. 19–38, 2022, doi: 10.1007/s13735-021-00218-1

[6] Z. Tu et al., "MAXIM: Multi-Axis MLP for Image Processing," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, LA, USA, 2022, pp. 5759-5770, doi: 10.1109/CVPR52688.2022.00568.

[7] J.-D. Tournier, R. Smith, D. Raffelt, R. Tabbara, T. Dhollander, M. Pietsch, D. Christiaens, B. Jeurissen, C.-H. Yeh and A. Connelly, "MRtrix3: A fast, flexible and open software framework for medical image processing and visualisation," *NeuroImage*, vol. 202, pp. 116137, 2019, doi: 10.1016/j.neuroimage.2019. 116137.

[8] A. P. Kusumah, D. Djayusman, G. R. Setiadi, A. C. Nugraha and P. Hidayatullah, “Counting Various Vehicles using YOLOv4 and DeepSORT,” *Journal of Integrated and Advanced Engineering (JIAE)*, vol. 3, no. 1, pp. 1-6, 2023, doi: 10.51662/jiae.v3i1.68

[9] P. Jiang, D. Ergu, F. Liu, Y. Cai and B. Ma, "A Review of Yolo Algorithm Developments," *Procedia Computer Science*, vol. 199, pp. 1066-1073, 2022, doi: 10.1016/j.procs.2022.01.135.

[10] X. Xu, M. Zhao, P. Shi, R. Ren, X. He, X. Wei and H. Yang, "Crack Detection and Comparison Study Based on Faster R-CNN and Mask R-CNN," *Sensors*, vol. 22, no. 3, pp. 1215, 2022, doi: 10.3390/ s22031215

[11] P. K. Sahoo, S. Mishra, R. Panigrahi, A. K. Bhoi and P. Barsocchi, "An Improvised Deep-Learning-Based Mask R-CNN Model for Laryngeal Cancer Detection Using CT Images," *Sensors*, vol. 22, no. 22, pp. 8834, 2022, doi: 10.3390/s22228834

[12] Z. Lyu, D. Zhang and J. Luo, "A GPU-free real-time object detection method for apron surveillance video based on quantized MobileNet-SSD," *IET Image Processing*, vol. 16, no. 8, pp. 2196-2209, 2022, doi: 10.1049/ipr2.12483

[13] M. Yunita, R. Muwardi and Z. Iklima, "Implementation of Bayesian inference MCMC algorithm in phylogenetic analysis of Dipterocarpaceae family," *SINERGI*, vol. 27, no. 1, pp. 23-30, 2023, doi: 10.22441/sinergi.2023.1.004
[14] X. Lu, J. Ji, Z. Xing and Q. Miao, "Attention and Feature Fusion SSD for Remote Sensing Object Detection," in IEEE *Transactions on Instrumentation and Measurement*, vol. 70, pp. 1-9, 2021, Art no. 5501309, doi: 10.1109/TIM.2021.3052575.
[15] L. Cheng, Y. Ji, C. Li, X. Liu and G. Fang, "Improved SSD network for fast concealed object detection and recognition in passive terahertz security images," *Scientific Reports*, vol. 12, no. 1, pp. 12082, 2022, doi: 10.1038/s41598-022-16208-0
[16] X. Lu, X. Kang, S. Nishide and F. Ren, "Object detection based on SSD-ResNet," *2019 IEEE 6th International Conference on Cloud Computing and Intelligence Systems (CCIS)*, Singapore, 2019, pp. 89-92, doi: 10.1109/CCIS48116.2019.9073753.
[17] K. Dong, C. Zhou, Y. Ruan and Y. Li, "MobileNetV2 Model for Image Classification," *2020 2nd International Conference on Information Technology and Computer Application (ITCA)*, Guangzhou, China, 2020, pp. 476-480, doi: 10.1109/ITCA52113.2020.00106.
[18] A. Sharma, J. Pathak, M. Prakash and J. N. Singh, "Object Detection using OpenCV and Python," *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, Greater Noida, India, 2021, pp. 501-505, doi: 10.1109/ICAC3N53548.2021.9725638.
[19] Y. Xiao, Z. Tian, J. Yu, Y. Zhang, S. Liu, S. Du and X. Lan, "A review of object detection based on deep learning," *Multimedia Tools and Applications*, vol. 79, pp. 23729–23791, 2020, doi: 10.1007/s11042-020-08976-6
[20] H. A. A. Osman and N. Z. Azlan, “Generating images for Supervised Hyperspectral Image Classification with Generative Adversarial Nets,” *Journal of Integrated and Advanced Engineering (JIAE)*, vol. 2, no. 2, pp. 107-112, 2022, doi: 10.51662/jiae.v2i2.80