IJEEC

International Journal of Electrical Engineering and Computing
Vol. 8, No. 2 (2024)

Short Report
UDC 004.439:621.395.721.5
DOI 10.7251/IJEEC2402068M

Cost vs. Performance: Raspberry Pi and ESP32 in Face Detection Tasks

Marko Malović[1], Elma Avdić[2] and Nikola Davidović[1]

[1] Faculty of Electrical Engineering, University of East Sarajevo, East Sarajevo, Bosnia and Herzegovina
[2]Dept. of IT, International Burch University, Sarajevo, Bosnia and Herzegovina

E-mail address: marko.malovic@etf.ues.rs.ba, elma.avdic@ibu.edu.ba, nikola.davidovic@etf.ues.rs.ba

Abstract— **A cost-effective, scalable solution that leverages low-cost embedded systems for computationally intensive tasks like face detection is critical for the growing demands of IoT applications, motivating the development of innovative hybrid architectures. This paper proposes such a framework, combining the ESP32 CAM as a distributed image capture unit with light preprocessing and the Raspberry Pi as a centralized processing unit. By addressing the limitations of standalone implementations—low computational capability of the ESP32 CAM and high deployment costs of the Raspberry Pi—our framework achieves a balance between affordability and performance. We present the analysis of the devices, evaluating metrics such as frame rate, detection accuracy, and cost-efficiency. The results highlight the potential of the hybrid system to significantly lower costs while enabling scalable, real-time face detection in IoT scenarios. This study contributes to the ongoing research by proposing an adaptable, resource-optimized framework suitable for diverse use cases, from smart surveillance to retail monitoring, paving the way for more efficient IoT-based vision systems.**

Keywords-component; Raspberry Pi, ESP32 CAM, face detection, microcontrollers, IoT, frame rate, detection accuracy, cost-efficiency

I. INTRODUCTION

In the last decade, the IoT (*Internet of Things*) industry has experienced significant growth. Since its first mention in scientific literature in 2006, IoT has become widely adopted across various sectors including healthcare, transportation and smart home technology [1]. Tasks like facial detection and recognition have become increasingly common. Facial processing has a number of applications in IoT including security cameras, smart lockers, smart robots as well as more specific approaches such as face mask detection [1-4]. This creates a demand for face detection solutions that are cost-effective and efficient.

Face detection is an important feature in diverse IoT applications such as surveillance and security monitoring, smart locks, automated attendance systems [4][7][9]. These tasks pose computational challenges, especially when implemented on resource-constrained hardware. Despite the progress in algorithms, IoT development in deploying face detection on embedded devices still presents a challenge as it asks for computationally and energy efficient, cost-effective hardware solutions [5].

The Raspberry Pi and ESP32 CAM are popular platforms for IoT development but their comparative performance in face detection tasks is underexplored. Raspberry Pi is a versatile single-board computer, offering computational power at a relatively high cost. The ESP32 CAM, in contrast, is a low-cost microcontroller with built-in camera capabilities, making it a candidate for cost-sensitive applications. Existing research of the topic of efficiency and implementation cost predominantly focuses on reducing CPU usage via more optimized algorithms [4-5] and reducing power consumption [5] while some of the researches prefer migrating image processing to cloud [6]. Real-world applications, however, often ask for on-device processing to minimize latency and also ensure privacy.

In this study, we aim to explore how do the Raspberry Pi and ESP32 CAM compare in terms of face detection performance under varying loads and under identical face detection tasks, the cost implications of deploying them, and if a hybrid system could leverage the strengths of both devices to optimize cost and performance for other, diverse IoT scenarios where we deal with resource-constrained environments.

The paper is structured as follows. Section II reviews related work in face detection on resource-constrained devices and their IoT applications. Section III describes the methods and experimental setup, including hardware and software configurations for microcontrollers and performance metrics. Section IV presents the results, focusing on frame rate, detection accuracy, and cost analysis. Section V discusses the implications of these findings for IoT applications and proposes a hybrid system design. Section VI concludes the paper outlining future research directions.

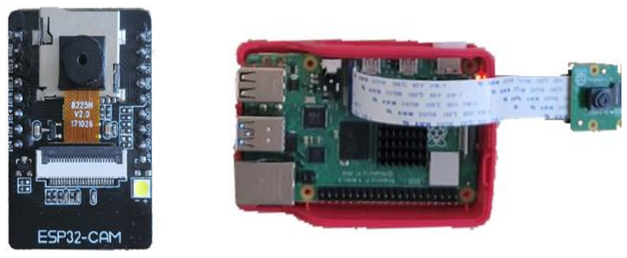

Figure 1. ESP32 CAM(left) and Raspberry Pi 4 model B(right)

II. Related Work

Face detection is a complex computational task which requires significant processing power and operating memory [5]. Such resources are more limited in IoT world than in traditional computing systems. Embedded devices like Raspberry Pi and ESP32 CAM provide a compact, cost-effective solution for real-world IoT applications but often struggle with computational demands.

Face detection encounters numerous challenges which can hinder accurate face capture. Factors such as eyewear, beard, specific head rotations and environmental conditions such as change in lighting can significantly impact detection accuracy [7]. To address these problems various face detection and recognition algorithms were developed, including Haar cascades, histogram of oriented gradients, support vector machine and other deep learning methods [8]. Various researches were conducted regarding optimal choice of the IoT devices depending on the power, cost, use case as well as performance [11-12]. There is also a distinction, even among the powerful high-performing microcontrollers, in regards of CPU (*Central Processing Unit*) type, existence of integrated GPU (*Graphical Processing Unit*) or presence of hardware-level parallelism and high customizability through the use of FPGA (*Field-Programmable Gate Arrays*) [12].

In this paper two microcontrollers were used, the ESP32 CAM and the Raspberry Pi 4 model B, both represented in Fig. 1. The ESP32 CAM proven practical in applications such as home security systems [9-10]. Due to the versatile GPIO (*General Purpose I/O*) capabilities, the face detection feature is often combined with sensors, usually motion sensors, while some of the researchers have also tried combination with fire sensors [9]. Additionally, the ESP32 CAM is used in low-cost vision-based tracked robots, where researchers simulated a military robot, which is able to recognize allies and foes thanks to ESP32 CAM's ability to perform facial detection and recognition [13].

The Raspberry Pi microcontroller is also widely used in the field of computer vision and facial detection applications [7][14][15]. Notable applications include smart system for emotion recognition [14], UAV with face detection system which had 80 – 98% success rate in facial recognition task, depending on the distance of the UAV from the ground [15]. The importance of choosing the most appropriate microcontroller for a project is significant in IoT world, especially given that the most of the IoT projects are designed with scalability and mass production in mind. Since facial processing is a computationally intensive task, it is important to choose the best combination of price and hardware to achieve the optimal results.

The research that has produced research paper has been financially supported by the European Union through the project Introducing Student Research Mobilities to BH Universities – INSTREAM, implemented by a consortium led by International Burch University, but do not necessarily represent the official position of the European Union or International Burch University and are the sole responsibility of the author.

III. Methodology

In this section, we describe the study conducted and the structure of the approach. First, we introduce the hardware configurations, software tools used, and the benchmarking process. Then, we describe our test dataset, evaluation metrics and the experimental setup design.

A. Hardware Configurations

For the experiment a Raspberry Pi 4 Model B was used, a versatile single-board computer equipped with 1.8 GHZ ARMv8 quad-core Cortex-a72 processor with 4GB of RAM ensuring robust computational capacity [17]. The rich input/output interface includes four USB ports, a gigabit Ethernet port and dual micro-HDMI outputs. For taking images a Raspberry Pi Camera Module 2 was connected via Raspberry Pi camera interface. This camera supports recording high-definition video at resolutions of 1080p at 60 frames per second and 720p at 60 frames per second, as well as capturing still images at 8-megapixel resolution [18].

This experiment also included ESP32 CAM microcontroller with OV2640 camera chip. The ESP32 CAM microcontroller, developed by AIThinker, features an integrated Wi-Fi antenna which enables remote access to the device over the internet, a feature that was used in the experiment. The device comes with 8MB of PSRAM which accelerates image processing and enhances overall efficiency. For programming the ESP32 CAM an FTDI programmer was used. The OV2640 camera chip is a low-voltage CMOS image sensor capable of capturing images in various formats, including full-frame, sub-sampled, scaled and windowed images [19].

B. Software and Algorithms

The Raspberry Pi platform supports many operating systems, predominantly Linux-based distributions. In this study, Ubuntu Server 22.04 LTS 64-bit was selected for optimal performance, given that it is a lightweight, console-only distribution that conserves resources compared to distributions with a GUI (*Graphical User Interface*). The face detection script was developed in Python programming language, using official Python IDE (*Integrated Development Environment*) with OpenCV as the core library. OpenCV was used for both, capturing images and analyzing frames using Haar Cascade classifier. This classifier is a set of pretrained cascades which can be used for efficient real-time facial detection [16]. Raspberry Pi was connected to the internet to enable remote access via Secure Shell.

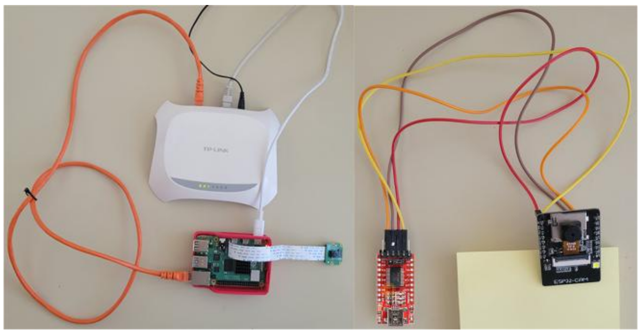

Figure 2. Microcontrollers with accompanying wiring and network equipment

The ESP32 CAM module used a customized version of the camera web server script developed by Espressif. This script was written in C programming language and it was customized and debugged using Arduino IDE. The script enables access to the device's video stream over the internet and includes face detection feature that can be dynamically activated or deactivated during the stream. Fig. 2 shows both microcontrollers with power adapters, jumper wires and router which enables internet access for both of the devices.

C. Dataset

A custom dataset comprising of 10 collages of images of various individuals was created to simulate real-world scenarios. Each collage varied in the number of faces (1 to 10) and included variations in lighting, face orientation, and distance to evaluate detection. The collages were displayed on a high-resolution monitor to ensure uniform input conditions.

D. Metrics

To evaluate the two devices for face detection tasks, we applied a structured approach including consistent lighting, resolution, and input image complexity to standardize our experimental conditions and ensure comparable results across both devices. Three key metrics were quantified to evaluate the performance of each device: frame rate (FPS, measured as the number of frames processed per second), detection accuracy (percentage of correctly identified faces in each collage, evaluated against a manually labelled ground truth for each image), and hardware stability (assessed as the device's ability to maintain consistent performance across varying workloads and image complexities).

E. Experimental Design

The experimental procedure was as follows. The measurements were obtained by positioning each of device's camera in the direction of the screen which displayed the dataset. Camera angles and distances were adjusted to ensure full screen coverage for consistent input across tests. Both devices, Raspberry Pi and ESP32 CAM, captured data from each collage for a duration of 5 seconds, during which the speed of detection, face detection accuracy, and the number of processed frames were measured. The Raspberry Pi Camera captured high-resolution images at 3280 × 2464 pixels while the ESP32 CAM was limited to 320x240 pixels as higher resolution settings face detection resulted in unreliable readings. For some cases, ESP32 CAM's resolution was further reduced to 240x240 pixels to maintain stable face detection results.

TABLE I. RASPBERRY PI AND ESP32 CAM COST COMPARISON

Raspberry Pi		**ESP 32 CAM**	
Part	***Price (EUR)***	***Part***	***Price (EUR)***
Raspberry Pi 4 model B	60.00	FTDI adapter	4.00
RPi Camera Module 2	27.00	Jumper wires	0.40
Power Supply	9.50	ESP32 CAM microcontroller	12.50
SanDisk SD Card	7.50	-	-
Board Case	8.00	-	-
TOTAL	122.00	-	16.90

F. Cost Comparison

The cost disparity between the ESP32 CAM and Raspberry Pi setups is significant and it occurs from their different design purposes. The ESP32 CAM is a compact microcontroller optimized specifically for image capture and video streaming. By contrast, the Raspberry Pi is a performant general-purpose microcontroller capable of supporting a wide range of applications beyond video and image processing. However, configuring Raspberry Pi for face detection requires additional components, such as a compatible camera module and an SD card along with proper DC adapter and board case, which substantially increase its overall setup cost. In comparison, the ESP32 CAM includes a built-in camera module, making it a compact, low-cost solution for specific tasks like video streaming. Table 1 presents a detailed cost breakdown for each setup, illustrating the ESP32 CAM's affordability versus the Raspberry Pi's flexibility which comes at a higher price point.

The cost disparity is significant: an ESP32 CAM setup costs €16.90 with included additional components such as jumper wires and adapter, while a Raspberry Pi face detection setup costs €122.00, roughly 7 times the cost of the ESP32 CAM setup. The most expensive Raspberry Pi parts are the microcontroller itself, which costs €60.00 and camera which comes at a price tag of €27.00 In comparison, the most expensive part of the ESP32 CAM is also the microcontroller, which comes with preinstalled camera, and it costs only €12.50. All prices were acquired from the official distributors of the ESP32 CAM by AIThinker and Raspberry Pi.

IV. RESULTS AND DISCUSSION

A. Data Presentation and Discussion

During the experiment, the Raspberry Pi microcontroller demonstrated superior performance compared to the ESP32 CAM. The most obvious distinction is in computational efficiency where Raspberry Pi processed data is, on average, 52% faster than the ESP device.

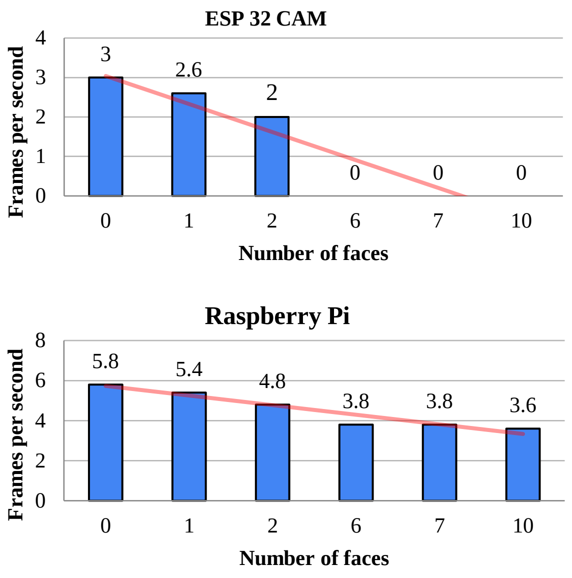

Figure 3. Number of faces and FPS for ESP32 CAM(top) and Raspberry PI (bottom)

This significant difference in processing speed highlights the Raspberry Pi’s capacity to handle intensive computational tasks more efficiently, while the ESP32 CAM showed limitations, particularly under higher processing demands. Fig. 3 shows differences in computational speeds between the two devices. For the ESP32 CAM, frame rate drops significantly as more faces appear in the image. Starting at 3 FPS (*Frames Per Second*) with zero faces, the rate decreases to 2.6 FPS with one face, and further to 2 FPS with two faces. Beyond two faces, the ESP32 CAM fails to process any frames successfully. In comparison, the Raspberry Pi demonstrates a more gradual decline in performance. It begins with a higher frame rate of 5.8 FPS with zero faces, slowly decreasing to 3.6 FPS as more faces are detected.

Table 2 provides details about frame rate values for each test case, alongside the percentage difference in performance between the two devices. The first case, which serves as the reference case, involved an image with no human faces. In this reference case, the Raspberry Pi achieves 5.8 FPS, while the ESP32 CAM reaches 3 FPS, resulting in a 48.28% performance difference. In the second case, where one face is present, both devices experience increased processing demands, leading to a slight reduction in speed.

TABLE II. FRAMES PER SECOND

Number of faces	**ESP32 FPS**	**Raspberry FPS**	**Difference (%)**
0	3	5.8	48.28%
1	2.6	5.4	51.85%
2	2	4.8	58.33%
6	0	3.8	100.00%
7	0	3.8	100.00%
10	0	3.6	100.00%

In this case Raspberry Pi maintained speed of 5.4 FPS while ESP32 CAM slowed to 2.6 FPS, increasing the performance difference to 51.85%. As number of faces increased with each test case, computational speeds went down. The ESP32 CAM stopped detecting faces after third test case. Probable cause of this behavior is low resolution of the camera. The Raspberry Pi continued to detect faces successfully and it maintained a consistent processing speed of 3.8 FPS for images containing 6 and 7 human faces. In the final test case, which included 10 human faces, the Raspberry Pi achieved a processing speed of 3.6 FPS. The Raspberry Pi device performed 52.82 percent better than ESP32 CAM in the first three test cases.

B. *Implications and Findings*

The ESP32 CAM, while cost-effective and functional for basic video streaming and image capture, demonstrated limited usage capabilities in domain of face detection. Low processing speeds limit the suitability of this device for application where consistent, high-speed, high image quality processing is needed. However, it remains a feasible choice in cases where speed and resolution are not primary concerns.

The Raspberry Pi demonstrated significantly better performance, maintaining stable frame rates and reliable operation even in cases with larger number of detected faces. This capability makes the Raspberry Pi a suitable candidate for more demanding IoT applications, such as video surveillance, visitor counting and access control systems, where higher processing power is needed. While the Raspberry Pi setup for face detection is considerably more expensive than the ESP32 CAM, it offers greater flexibility in terms of operating system choices, software compatibility and robustness, justifying its use in applications where performance outweighs cost.

C. *Limitations of the Study*

There are multiple versions of ESP32 CAM available and some are more performant than others, depending on the use case. In this experiment the ESP32 CAM by AIThinker was used. However, the better pick for this experiment would be ESP32 CAM S3 since it has better face detection performance since it comes with much performant processor, but it was unavailable for purchase at the time of doing this research.

Fig. 4, addresses the limitations identified in standalone ESP32 CAM deployment, in particular its constrained processing power and inability to handle complex tasks reliably.

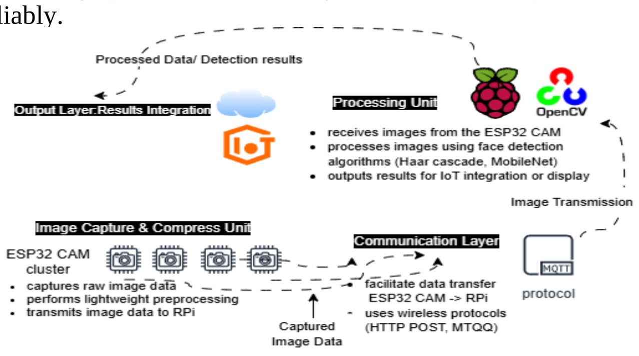
Processed Data/ Detection results
Output Layer:Results Integration
IoT
Processing Unit
OpenCV
receives images from the ESP32 CAM
processes images using face detection algorithms (Haar cascade, MobileNet)
outputs results for IoT integration or display
Image Transmission
Image Capture & Compress Unit
ESP32 CAM cluster
captures raw image data
performs lightweight preprocessing
transmits image data to RPi
Communication Layer
facilitate data transfer ESP32 CAM -> RPi
uses wireless protocols (HTTP POST, MTQQ)
Captured Image Data
MQTT
protocol

Figure 4. Hybrid model

TABLE III. SCALABLE, COST-EFFECTIVE FACE DETECTION IN IOT APPLICATION

Hybrid component	Role	Strengths leveraged	Weaknesses addressed
ESP32 CAM	Image capture	Low cost, compact size, wireless connectivity	Limited processing capability and accuracy
Comm Module	Data transfer	Supports lightweight wireless protocols (MQTT, HTTP)	Mitigates latency through efficient transmission
Raspberry Pi	Processing and face detection	Computational power, diverse software support	High cost for standalone large-scale system
Out Module	Results integration and display	Centralized processing by Raspberry Pi + higher-level insights (e.g., total detected faces, alerts for anomalies) instead of raw data	Inaccuracies in data upstream from ESP32 CAM

This architecture integrates the ESP32 CAM as an image capture unit doing lightweight preprocessing of data (e.g., resizing and compression) to reduce the bandwidth required for the next step of data transmission. It transmits the compressed image data to the Raspberry Pi device through the communication module using wireless protocols (HTTP, MQTT). The complex processing is done in central processing unit represented by the Raspberry Pi, where face detection algorithms are deployed (Haar cascades or MobileNet [16] [19]) and then outputs actionable insights to the output layer where we can visualize the results or integrate them into other IoT systems.

In this approach, we balance out the affordability and scalability of ESP32 CAM with the Raspberry Pi computational power, making it fit for diverse IoT scenarios such as real-time surveillance, attendance systems, and low-cost retail monitoring. The proposed architecture leverages the strengths of both devices, addressing their weaknesses, as shown in Table 3, offering a cost-effective and scalable solution for IoT-based face detection applications.

D. Future Research Directions

Future research may include testing older versions of Raspberry Pi in the case of face detection, and comparing them to the ESP32 CAM results. Older versions of the Raspberry controller are cheaper but less performant due to the hardware limitations. However, the ESP32 CAM also has different versions, some more performant than others, so a detailed comparison between all of the devices would be a very interesting and useful research topic.

V. CONCLUSION

This study provides a detailed comparison of the two microcontrollers in the case of face detection, analyzing their performance, cost-effectiveness, and applicability in IoT projects. The results of the experiment demonstrated that the Raspberry Pi outperformed the ESP32 CAM in nearly every aspect of face detection. It consistently maintained higher frame rates and showed robust performance even with increasing numbers of faces. However, these advantages come with a significantly higher cost, making the Raspberry Pi an expensive option for IoT applications where cost is a critical factor.

However, the ESP32 CAM showed significant limitations in terms of processing speed, resolution, and stability. Its inability to handle images containing multiple faces indicate that it is not suitable for high-performance face detection applications. This microcontroller is considerably more affordable than the Raspberry Pi, which has the potential to make it a viable choice for cost-sensitive IoT applications.

In summary, the ESP32 CAM may be preferred in low-cost scenarios with minimal processing demands, whereas the Raspberry Pi is better suited for applications that prioritize speed, accuracy, and reliability in face detection tasks. The choice between the two microcontrollers ultimately depends on the specific requirements of the intended application and the balance between budget constraints and performance needs.

ACKNOWLEDGMENT

The research that has produced research paper has been financially supported by the European Union through the project Introducing Student Research Mobilities to BH Universities – INSTREAM, implemented by a consortium led by International Burch University, but do not necessarily represent the official position of the European Union or International Burch University and are the sole responsibility of the author.

REFERENCES

[1] Dachyar, M., Zagloel, T. Y. M. and Saragih, L. R. (2019) 'Knowledge growth and development: internet of things (IoT) research, 2006–2018', Heliyon, 5(8).

[2] Aydin, I. and Othman, N. A. (2017) 'A new IoT combined face detection of people by using computer vision for security application', 2017 International Artificial Intelligence and Data Processing Symposium (IDAP). IEEE.

[3] Naseri, R. A. S., Kurnaz, A. and Farhan, H. M. (2023) 'Optimized face detector-based intelligent face mask detection model in IoT using deep learning approach', Applied Soft Computing, 134, 109933.

[4] Mostakim, N., Sarkar, R. R. and Hossain, M. A. (2019) 'Smart locker: IoT based intelligent locker with password protection and face detection approach', International Journal of Wireless and Microwave Technologies, 9(3), pp. 1–10.

[5] Choi, K. and Sobelman, G. E. (2021) 'Optimized face detection and alignment for low-cost and low-power IoT systems', 2020 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS). IEEE.

[6] Li, S., et al. (2019) 'Optimization of Face Recognition System Based on Azure IoT Edge', Computers, Materials & Continua, 61(3).

[7] Balogh, Z., Magdin, M. and Molnár, G. (2019) 'Motion detection and face recognition using raspberry pi, as a part of, the internet of things', Acta Polytechnica Hungarica, 16(3), pp. 167–185.

[8] Dubovečak, M., Dumić, E. and Bernik, A. (2023) 'Face detection and recognition using raspberry PI computer', Tehnički glasnik, 17(3), pp. 346–352.

[9] Cahyono, F. Y. A., Suharto, N. and Mustafa, L. D. (2022) 'Design and build a home security system based on an esp32 cam microcontroller with telegram notification', Journal of Telecommunication Network (Jurnal Jaringan Telekomunikasi), 12(2), pp. 58–64.

[10] Salikhov, R. B., Abdrakhmanov, V. Kh. and Safargalin, I. N. (2021) 'Internet of things (IoT) security alarms on ESP32-CAM', Journal of Physics: Conference Series, 2096(1).

[11] Chen, D., et al. (2016) 'Platform choices and design demands for IoT platforms: cost, power, and performance tradeoffs', IET Cyber - Physical Systems: Theory & Applications, 1(1), pp. 70 - 77.

[12] Ganguly, P. (2016) 'Selecting the right IoT cloud platform', 2016 International Conference on Internet of Things and Applications (IOTA), Pune, India, pp. 316–320. doi: 10.1109/IOTA.2016.7562744.

[13] Budiharto, W., et al. (2022) 'Low-cost vision-based face recognition using ESP32-CAM for tracked robot', ICIC Express Letters Part B: Applications, 13(3), pp. 321–327.

[14] Suja, P. and Tripathi, S. (2016) 'Real-time emotion recognition from

[15] facial images using Raspberry Pi II', 2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN). IEEE.

[16] Daryanavard, H. and Harifi, A. (2018) 'Implementing face detection system on UAV using Raspberry Pi platform', *Electrical Engineering (ICEE), Iranian Conference on*. IEEE

[17] OpenCV(n.d.) 'Face Detection using Haar Cascades', *OpenCV Documentation*. Available at: https://docs.opencv.org/4.x/d2/d99/tutorial_js_face_detection.html (Accessed: 5 November 2024)

[18] Raspberry Pi Documentation (n.d.) 'Raspberry Pi Specification'. Available at: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/ (Accessed: 26 November 2024).

[19] Raspberry Pi Documentation (n.d.) 'Camera Module V2 Specification'. Available at: https://www.raspberrypi.com/products/camera-module-v2/ (Accessed: 26 November 2024)

[20] Components101 (n.d.) 'ESP32-CAM Camera Module', *Components101 Documentation*. Available at: https://components101.com/modules/esp32-cam-camera-module (Accessed: 26 November 2024)

Marko Malović is a teaching assistant at the Faculty of Electrical Engineering, University of East Sarajevo. He is also a student on the master's program at the same faculty. His research interests focus on software related topics including databases, IoT and web-based applications

Nikola Davidović works as an Assistant Professor at the Faculty of Electrical Engineering, University of East Sarajevo. He obtained his Ph.D. in Technical Sciences in the field of Computer Science from the Faculty of Electrical Engineering, University of East Sarajevo. His areas of expertise include computer architecture, hardware and firmware programming.

Elma Avdic is a lecturer at the Department of Information Technology, International Burch University, Sarajevo. She received her Ph.D. from Trinity College Dublin in 2019, in the area of radio spectrum sharing in future wireless networks. Her current research interests are focused on sustainable, data-driven and intelligently regulated networked systems.