energies

MDPI

Article

Power Requirements Evaluation of Embedded Devices for Real-Time Video Line Detection

Jakub Suder *, Kacper Podbucki and Tomasz Marciniak

Division of Electronic Systems and Signal Processing, Institute of Automatic Control and Robotics, Poznan University of Technology, Piotrowo 3A, 60-965 Poznan, Poland; kacper.podbucki@put.poznan.pl (K.P.); tomasz.marciniak@put.poznan.pl (T.M.)

* Correspondence: jakub.suder@put.poznan.pl

Abstract: In this paper, the comparison of the power requirements during real-time processing of video sequences in embedded systems was investigated. During the experimental tests, four modules were tested: Raspberry Pi 4B, NVIDIA Jetson Nano, NVIDIA Jetson Xavier AGX, and NVIDIA Jetson Orin AGX. The processing speed and energy consumption have been checked, depending on input frame size resolution and the particular power mode. Two vision algorithms for detecting lines located in airport areas were tested. The results show that the power modes of the NVIDIA Jetson modules have sufficient computing resources to effectively detect lines based on the camera image, such as Jetson Xavier in mode MAXN or Jetson Orin in mode MAXN, with a resolution of 1920 × 1080 pixels and a power consumption of about 19 W for 24 FPS for both algorithms tested.

Keywords: microprocessor power modes; embedded systems; DVFS; NVIDIA Jetson; Raspberry Pi; video analysis; line detection

Citation: Suder, J.; Podbucki, K.; Marciniak, T. Power Requirements Evaluation of Embedded Devices for Real-Time Video Line Detection. *Energies* **2023**, *16*, 6677. https://doi.org/10.3390/en16186677

Academic Editors: Wiseman Yair, Mariusz Orlikowski and Krzysztof Górecki

Received: 28 July 2023
Revised: 6 September 2023
Accepted: 13 September 2023
Published: 18 September 2023

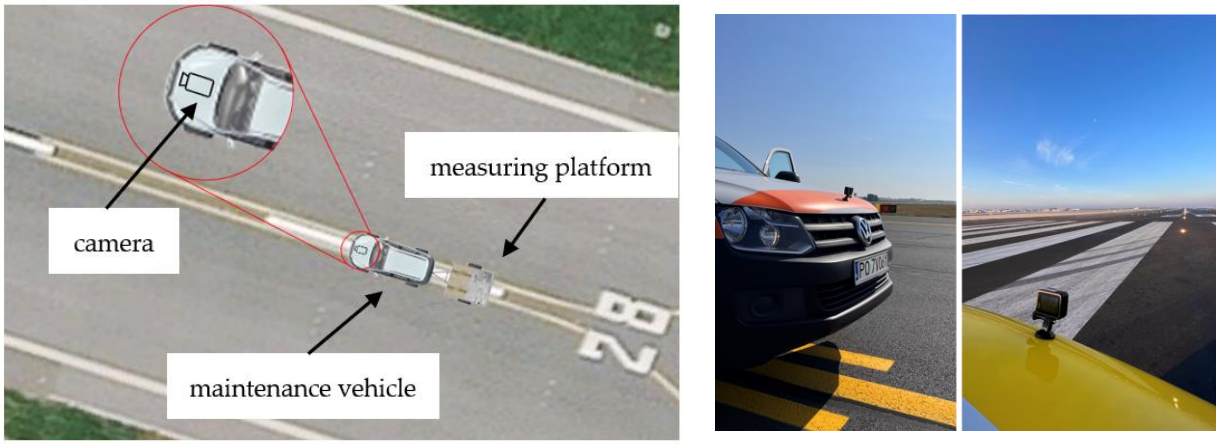
1. Introduction

Modern embedded systems give the possibility of effective video sequence processing in real time. Algorithms for video sequence analysis can be based on classic image processing methods and on neural networks, the use of which is becoming more and more popular. Cameras are installed in cars and autonomous vehicles [1,2], for city monitoring, tracking [3], but also in specialized applications, such as inspection vehicles [4]. Such systems help operators to observe urban spaces or machine surroundings and to supervise and respond to current events. In some cases, they are able to completely eliminate the human factor, automatically issuing specific commands to other systems. In the use of autonomous vehicles in airport areas [4], the accuracy of the algorithms and the hardware performance of the devices are extremely important, because it is possible to ensure safety. Moreover, such vehicles should be extremely energy efficient due to the large areas of the airports in which they are moving and have the necessary energy reserve in the event of emergencies.

For this reason, it is necessary to control the lighting and navigation equipment installed in the airport areas, with particular emphasis on the runway center line, the touchdown zone, and the taxiway center line. To perform the correct drive over the tested lamp, due to the width of the starting road and the lack of static reference points, it is necessary to prepare a system that will allow the operator/driver to carry out the measurement correctly [4]. Such systems are more often based on vision solutions. Therefore, the task of detecting runway lines and markings plays an important role in ensuring the safe movement of aircrafts on the ground [5]. They can also be helpful in determining the position of the measurement platform during inspections, because the position of the lamps is closely related to the location of horizontal markings on airport surfaces [6].

Processing video sequences requires the use of computing units of different power [7,8], depending on the selected algorithm and the operations performed on the image [5,9]. In mobile applications, it is important to find a compromise between the compact size of the

system, power consumption, and computer performance. To select the best equipment that can be used in individual cases, it is necessary to perform a comparative analysis of the available embedded systems, which, due to their small size and relatively reduced energy demand, can be installed in mobile autonomous vehicles, such as a measurement platform for quality testing of airport lamps. The need to build such a platform and control airport lamps came from the growing requirements of aviation safety agencies, which introduce more and more stringent regulations regarding the need to control airport lighting [6]. Lamps located in airport areas have a very important role in air navigation, because their visibility determines the possibility of performing individual air operations. The decrease in their light efficiency, and thus the need for cyclical checking, results from the use of halogen bulbs or LEDs but also from the winter season. During this period, airport areas are cleared of snow using ploughs with metal brushes. This causes tarnishing and scratching of the prisms of the in-pavement lamps, and even creates cracks, which exclude the lamp from further operation. Figure 1a shows the idea of a measurement platform in a real working environment (on the runway), Figure 1b shows the camera mounted on the airport maintenance vehicle, and Figure 1c shows the operator view of the camera on the airport maintenance vehicle located on the runway.



(**a**) (**b**) (**c**)

Figure 1. Measuring platform for quality testing of airport lamps (**a**), camera mounted on the airport maintenance vehicle (**b**), camera on the airport maintenance vehicle located on the runway (**c**).

In order for the platform to properly ride over the lamp built into the runway, it is necessary to use a vision system that correctly identifies the markings used on the runway and other airport areas and makes the necessary correction of the trajectory to improve the measurement. The specificity of a runway, which is much wider than public roads, makes it more difficult to find the right reference point because the edges of the pavement are often outside the frame. The principle of operation of the vision system on the measuring platform for quality testing of airport lamps is shown in Figure 2.

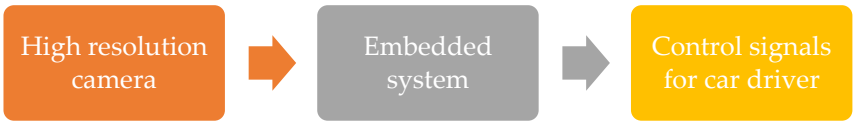


Figure 2. Block scheme of airport line tracking.

The purpose of the research was to analyze the performance of the video processing rate in terms of power requirements of the Raspberry Pi 4 and NVIDIA Jetson platforms. The tests carried out were aimed at estimating the energy consumption for the operating modes defined by the manufacturer. This manuscript is a revised and extended version

of an article presented at the 29th International Conference Mixed Design of Integrated Circuits and Systems, held in Wroclaw, Poland, from 23 to 25 June 2022 [10].

2. Related Works

The development of microcontrollers enables the implementation of more and more advanced image processing algorithms and video sequences. Because of the low energy consumption, it is possible to build portable devices powered by batteries. The design of prototypes of mobile vision processing solutions is facilitated by a wide range of minicomputers from the Raspberry Pi and NVIDIA Jetson families. It should be noted that industrial versions can also be used for these solutions, which offer professional performance and are resistant to long-term operation [11].

An important aspect in the design of such devices is the assessment of computational efficiency, which in the case of video sequence analysis is related to image resolution and the number of frames per second. The second important factor is power consumption, which should be as low as possible to reduce the size of the device (mainly the number and weight of the batteries) and reduce the need for frequent charging.

In general, the manufacturers of evaluation platforms present the performance of the microprocessors used, and on specialized websites (for example, on the [12] website), you can find comparisons of such metrics as integer math, floating point math, find prime numbers, random string sorting, data encryption, and data compression. Such metrics are difficult to relate directly to image processing performance, especially if the algorithm contains conditional instructions.

Assessment of computational performance can also be found on the [13] website; the manufacturer focuses on popular deep neural network solutions. Performance results in the form of samples/sec are presented for various NVIDIA Jetson models taking into account power requirements. It should be noted that the indicated neural networks have relatively low resolutions and the consumption analysis does not take into account the various possible power supply modes. For example, we can find the information that each NVIDIA Jetson module was run with the maximum performance MAXN.

The issue of video sequence processing efficiency is also discussed in scientific publications. In article [14], various algorithms such as the canny edge detection algorithm, road line tracking, face and eye recognition, motion detection, and object detection are discussed. Raspberry Pi 4 was indicated as the experimental platform, for which maximum power and frequency values of the CPU and GPU were given, but the exact values for individual algorithms were not indicated. The Raspberry Pi 4 module was also used in the research presented in the article [15]. The author focused on the frame rate, frame transfer delay, and frame processing time. The energy consumption aspect has not been investigated.

The results of video sequence processing performance analysis can also be found in publications for NVIDIA Jetson family modules. The paper [16] discusses a vision system to recognize fiducial markers including ARTag type. The vision system consists of two Logitech HD Pro Webcam C920 cameras and a NVIDIA Jetson TX2 module that performs digital image processing. NVIDIA Jetson Orin AGX performance and power consumption analyses are presented in article [17]. The mean Average Precision (mAP) was tested as a function of FPS (frames per second) and the different image sizes. The line detection task with the use of NVIDIA Jetson Xavier NX is discussed in [18]. The authors proposed a CNN Encoder–Decoder network architecture and tested their solution at different image resolutions and an input image size up to 1280×720. In all the above papers, energy consumption aspects were not analyzed.

A benchmark analysis of NVIDIA Jetson Platforms (Nano, TX2, NX, and AGX) are shown in the paper [19]. Measurements of resource usage and power consumption are presented without the influence of image resolution. Performances of single-board computers in NVIDIA Jetson Nano, NVIDIA Jetson TX2, and Raspberry PI4 through the Convolutional Neural Network (CNN) algorithm created by using a fashion product images dataset are compared in [20]. The authors of this paper analyzed performance defined as the pro-

cessing power (Central Processing Unit—CPU, Graphics Processing Unit—GPU), memory (Random-Access Memory—RAM, cache), power consumption, and cost. Unfortunately, no analysis of frame rate speed processing (Frames Per Second—FPS) was conducted.

The experiments presented in the following sections were based on the ideas presented in our previous article [5]. Due to the potential of using such solutions in battery-powered devices, a comprehensive analysis of both the FPS performance and the power consumption for various operating modes of the device was performed.

3. Materials and Methods

3.1. Tested Platforms

These systems are often used in machine learning as central units of autonomous cars [21,22] and to detect events in the image [23]. As part of the work and tests, it was decided that four microcomputers were to be used and examined:

- Raspberry Pi 4B
- NVIDIA Jetson Nano
- NVIDIA Jetson Xavier AGX
- NVIDIA Jetson Orin AGX.

Figure 3 shows the embedded devices used during experimental tests.

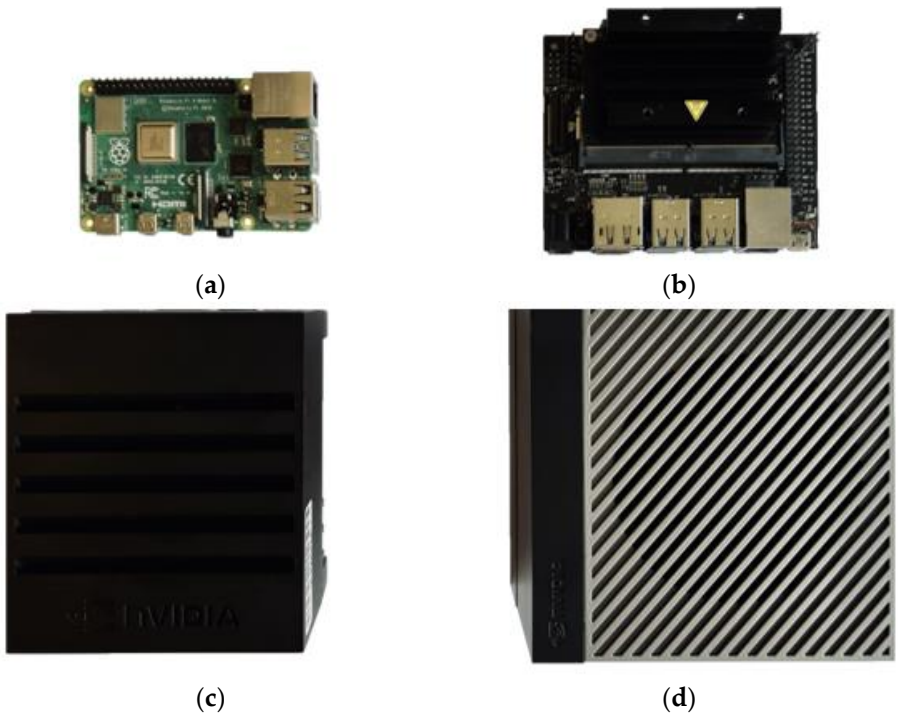


Figure 3. Embedded devices used during experimental tests: (**a**) Raspberry Pi 4B, (**b**) NVIDIA Jetson Nano, (**c**) NVIDIA Jetson Xavier AGX, and (**d**) NVIDIA Jetson Orin AGX.

The Raspberry Pi 4B is characterized by the Broadcom BCM2711 64-bit Quad-Core Advanced RISC Machine (ARM) Cortex-A72 processor clocked at 1.5 GHz and ARMv8-A architecture. The tested model is equipped with 8 GB of Low Power Double Data Rate 4 (LPDDR4) RAM. The device has only one power mode, and its specification is shown in Table 1.

The second device tested was an NVIDIA Jetson Nano microcomputer. This module was equipped with a quad-core ARM A57 processor clocked at 1.43 GHz and 4 GB of 64-bit LPDDR4 RAM memory. Compared to the Raspberry Pi 4B, NVIDIA Jetson Nano has a 128-core Maxwell GPU clocked at 921 MHz, which is a great advantage in the case of algorithms using machine learning. Depending on the neural network model used,

the performance of the NVIDIA Jetson Nano may be up to three times greater in this comparison [20,24]. In terms of the CPU processor, the NVIDIA Jetson Nano module is comparable in performance to the Raspberry Pi 4B microcomputer. This microcomputer offers two power modes: energy-saving (5W) and MAXN (10W). A summary is presented in Table 2 [25].

Table 1. Performance characteristics of the Raspberry Pi 4B 8GB.

Property	Mode
	-
Max power [W]	6
Online CPU	4
CPU max frequency [MHz]	1500
GPU TPC	4
GPU max frequency [MHz]	500

Table 2. Comparison of NVIDIA Jetson Nano operating modes.

Property	Mode	
	MAXN	**5 W**
Max power [W]	10	5
Online CPU	4	2
CPU max frequency [MHz]	1479	918
GPU TPC	1	1
GPU max frequency [MHz]	921.6	640
RAM max frequency [MHz]	1600	1600

The third microcomputer used was an NVIDIA Jetson Xavier AGX with an octa-core, 64-bit ARM®v8.2 processor with 8MB L2 and 4MB L3 cache, a 512-core NVIDIA Volta™ GPU and 64 Tensor cores, two NVIDIA Deep Learning Accelerator (NVDLA) engines, 32 GB of 256-bit LPDDR4x memory, and 32 GB of embedded Multi-Media Card (eMMC) memory. From an experimental point of view, it is interesting; it has 8 different power modes: MAXN, 10 W, 15 W, 30 W, 30 W 6CORE, 30 W 4CORE, 30 W 2CORE, and 15 W DESKTOP, which are summarized in Table 3 [25].

The fourth microcomputer, NVIDIA Jetson Orin AGX, the latest innovation from NVIDIA's embedded AI computing device lineup, represents a significant leap forward in terms of both performance and energy efficiency. This advanced AI computing platform is designed to facilitate the development and deployment of AI and robotics applications, providing high-performance computation in an energy-efficient, compact module. NVIDIA Jetson Orin AGX is based on the Orin system-on-a-chip (SoC), which harnesses the power of next-generation GPU architecture, advanced ARM CPU cores, and high-speed memory technology to deliver exceptional computational performance.

It delivers remarkable computational throughput, capable of performing multibillion operations per second (TOPS), providing a significant boost for machine learning and other data-intensive applications. This improvement is vital for applications that require real-time or near-real-time processing, such as autonomous vehicles, drones, and advanced robotics. Moreover, the Jetson Orin AGX supports a wide range of high-speed interfaces, including PCIe Gen 4 and USB4. These interfaces can be used to connect a variety of peripheral devices, such as sensors, cameras, and other input/output devices. Furthermore, it features Gigabit Ethernet and Wi-Fi capabilities, allowing for fast, reliable data transmission in networking applications. This unit is the most efficient built-in device in our comparison. This microcomputer also can work with four power modes: MAXN, 15 W, 30 W, and 50 W. The summary is presented in Table 4 [26].

Table 3. Comparison of NVIDIA Jetson Xavier AGX 32 GB operating modes.

Property	Mode							
	MAXN	**10 W**	**15 W**	**30 W**	**30 W 6CORE**	**30 W 4CORE**	**30 W 2CORE**	**15 W DESKTOP**
Max power [W]	n/a	10	15	30	30	30	30	15
Online CPU	8	2	4	8	6	4	2	4
CPU max frequency [MHz]	2265.6	1200	1200	1200	1450	1780	2100	2188
GPU TPC	4	2	4	4	4	4	4	4
GPU max frequency [MHz]	1377	520	670	900	900	900	900	670
RAM max frequency [MHz]	2133	1066	1333	1600	1600	1600	1600	1333

Table 4. Comparison of NVIDIA Jetson Orin AGX 64 GB operating modes.

Property	Mode			
	15 W	**30 W**	**50 W**	**MAXN**
Max power [W]	15	30	50	n/a
Online CPU	4	8	12	12
CPU max frequency [MHz]	1113.6	1728	1497.6	2201.6
GPU TPC	3	4	8	8
GPU max frequency [MHz]	420.75	624.75	828.75	1301
RAM max frequency [MHz]	2133	3200	3200	3200

3.2. Datasets

For experimental tests, we used a dataset of video sequences, obtained at Poznań–Ławica Airport. Data recording was done with two types of cameras: the GoPro Hero+ and GoPro Hero 8. The dataset consists of over 87 GB of video recordings containing images from the restricted area of the Poznań–Ławica Airport.

The GoPro Hero+ camera is resistant to changing external weather conditions and enables its appropriate and safe mounting as a piece of measuring platform system equipment. The GoPro Hero+ camera allows for recording images with a maximum resolution of 1920 × 1080 pixels at 60 frames per second. It has an 8-megapixel CMOS sensor and the aperture value is f/2.8. This camera uses L4.1 (H.264/MPEG-4 AVC) level coding. Thanks to the built-in WiFi module, it is also possible to transmit the video stream in real time. The GoPro Hero+ camera is equipped with an advanced image processing algorithm that ensures optimal image quality under various lighting conditions [27].

The GoPro Hero 8 is a high performance action camera designed to capture stable and high-quality footage in a variety of extreme environments [28]. In terms of resolution and framerate, the GoPro Hero 8 supports 4K video recording at up to 60 frames per second, 2.7K at up to 120 fps, and Full HD (1080 p) at up to 240 fps. The GoPro Hero 8 also features advanced image stabilization technology, which reduces camera shake to create smooth footage, even in high-motion scenarios. Still-image capture on the GoPro Hero 8 offers 12-megapixel resolution, with options for HDR, which enhances details in highlights and shadows. The technical specifications of the GoPro Hero 8, along with its ability to capture professional-quality video under a wide range of environmental conditions, make it a versatile tool for researchers and scientists looking to document their work in the field or in any challenging environment [29].

Figure 4 shows sample frames with the road markings used on airport areas in the form of the central axis of the runway (white dashed lines), as well as exit tracks and central axes of taxiways (solid yellow lines). It should be noted that the airport lamps are not mounted in the axis of the center lines, but with a certain shift; therefore, it is so important

to correctly detect the edges of the airport markings in order to determine the relative position of the next luminaire and to correct the approach trajectory.

Figure 4. Sample frames from recordings made at Poznań–Ławica Airport.

3.3. Line Detection Algorithms

As mentioned at the end of Section 2, two algorithms implementing a set of image processing operations, developed by the authors in previous publications [5], were used for experimental research. In this paper, analyzed algorithms process the input resolution of the video sequence without resizing. The following is a synthetic description of their operation.

3.3.1. Image Segmentation in the HSV Color Space

This algorithm (Figure 5) was implemented in Python 3.7.6 using the OpenCV library (version 4.2.0). The main innovation involves isolating color shades of lines (white or yellow) from the runway's grey or dark grey background. The algorithm consists of several stages:

1. Color Space Adjustment: The color space is transformed from RGB to HSV. This is done to prevent distortions due to uneven lighting that could affect the color scale.
2. Color Isolation: Shades of white and yellow are isolated using predefined saturation thresholds and the power of the white light component. The resulting mask identifies detected colors with a pixel value of 255 (white), while the rest of the frame has a value of 0 (black).
3. Edge Detection: This mask can then be used as input to the Canny algorithm for edge detection.

The algorithm has dedicated functions for color separation, Region of Interest (ROI) determination, line detection, and supervision. Given the wide-angle view of the camera, which, although beneficial for capturing the entire runway width, may include irrelevant elements causing interference, ROI is set manually. This ROI covers the central part of the image (where central lines should be located) plus a margin for error, excluding horizontal lines near the horizon. Post-processing, the image undergoes a Hough transform for line detection, specifically the Probabilistic Hough Line Transform. The final output is the original image with lines drawn along the edges of the stripes painted on the airport areas.

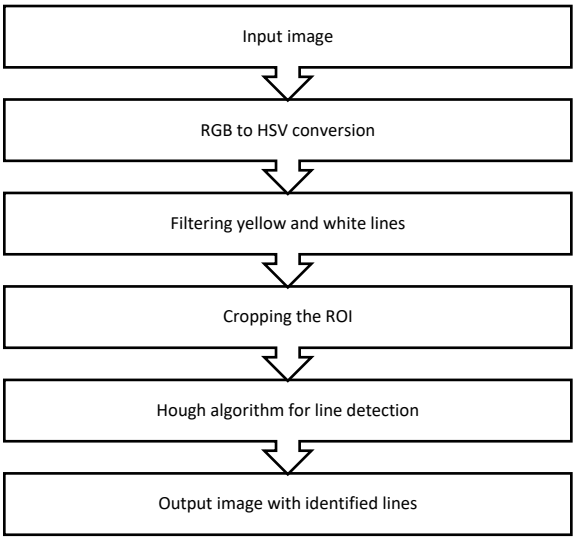


Figure 5. Block diagram of line detection with image segmentation in the HSV color space.

In particular, this algorithm effectively detects both white and yellow lines simultaneously due to the use of a different color space. The mask for yellow and white is calculated separately and then added. This approach uniquely enables efficient line detection within airport areas.

3.3.2. Line Detection Using the Hough Algorithm with Scharr Mask Filtering

The study discusses an algorithm designed to improve line detection quality and precision in images, specifically focused on airport runway markings. The algorithm (Figure 6) is implemented in Python 3.7.6 using the OpenCV library (version 4.2.0) and operates in several key stages:

1. Noise Removal: Noise in the image is removed using the median filter, which does not distort visible edges, which is crucial for line detection.
2. Image Sharpening: The image is then sharpened using a specific kernel, enhancing the visibility of the edges.
3. Color Space Transformation: The image is transformed from RGB space to greyscale.
4. Contrast Enhancement: Due to the wear and tear on airport runway lines and associated low contrast, histogram equalization is used to emphasize less visible details.
5. Edge Detection: The Canny algorithm is then applied for edge detection, with parameters set to minVal = 400 and maxVal = 500.
6. Vertical Edge Isolation: To isolate vertical edges crucial for centerline detection, all other edges detected by the Canny algorithm are discarded using a Scharr mask with a specific kernel array.
7. Line Detection: Probabilistic Hough Line Transform is used for line detection, resulting in an output frame with central axis lines superimposed on the original image.

The algorithm also incorporates an autonomous method of road marking recognition, based on the Support Vector Machine (SVM), to mitigate the effects of external factors such as viewpoint, brightness, and background. Experimental testing demonstrated a recognition accuracy of more than 97% with a frame time of 0.26 s. The algorithm also proved effective for identifying runway sign positions, achieving an accuracy of 95.1% for runway sign detection and 89.2% accuracy for identifying wait positions. These results suggest that the algorithm can support safety efforts in complex and large airports, offering a valuable complement to pilot education and ground radar systems.

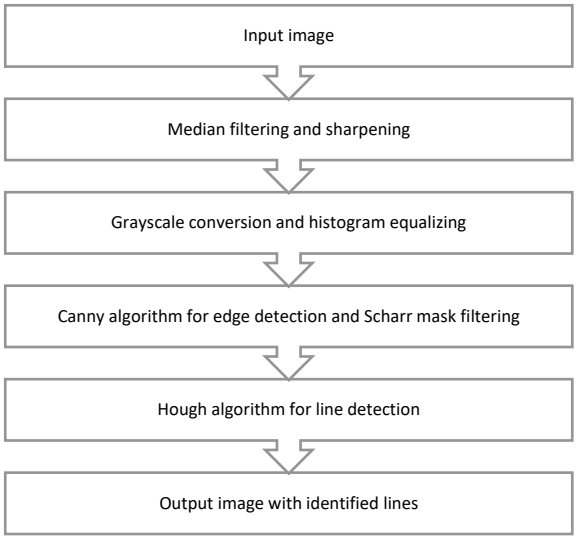


Figure 6. Block diagram of line detection using the Hough algorithm with Scharr mask filtering.

4. Results and Discussion

4.1. Performance Comparison of Embedded Systems

According to Section 3, analysis of algorithm performance was performed using Raspberry Pi 4B, NVIDIA Jetson Nano, NVIDIA Jetson Xavier AGX, and NVIDIA Jetson Orin AGX modules. The aim of the experiment was to segment the lines along the runway of the airport to determine the correct path. The video sequences used show the airport areas with horizontal markings that were registered at the Poznań–Ławica Airport.

Tests were carried out in 10 s fragments, selected randomly from the entire video sequence. Additionally, each fragment was scaled to six resolutions: 1920 × 1080 (Full HD), 1600 × 900 (HD +), 1366 × 768 (HD), 1280 × 720 (WXGA), 640 × 360 (nHD), and 320 × 180. Figures 7 and 8 show a comparison of the mean number of frames analyzed per second that were obtained during the processing of 10 s fragments depending on their resolution and the embedded system used for individual algorithms. It should be noted that the power consumption is due to the specificity of the image processing algorithm; thus, the process of randomizing the sequences was performed only once and then submitted for offline testing.

During all experiments for different platforms and for different modes of operation, identical software (without any specific modification for particular hardware) using CPUs only was used. Tests were carried out using different power consumption modes.

The algorithm based on image segmentation in the HSV color space, due to its lower computational complexity [5], is characterized by shorter processing times than the line detection using the Hough algorithm with Scharr mask filtering. The duration processing time of the program increases with the higher input resolution. This relationship is visible for each embedded device tested and for the selected power consumption mode. The Raspberry Pi 4B microcomputer allowed almost identical results as in the case of the NVIDIA Jetson Nano module in MAXN power mode. In the case of NVIDIA Jetson Xavier AGX, the increase in the number of processed frames per second on the maximum clock speed of the CPU processor can be observed. As shown in Table 3, higher processor timings are obtained for higher power consumption modes. For the 30 W mode, depending on the number of processor cores used, the maximum clock frequency also changes: the fewer active cores, the higher frequency. A similar dependence can be seen for NVIDIA Jetson Orin AGX, recommended in Table 4, where the clock frequency of the processor also changes. In the case of the most efficient units and this algorithm, NVIDIA Jetson Orin AGX and NVIDIA Jetson Xavier AGX, it is possible to process video sequences in real time at maximum resolution, obtaining values over 24 FPS.

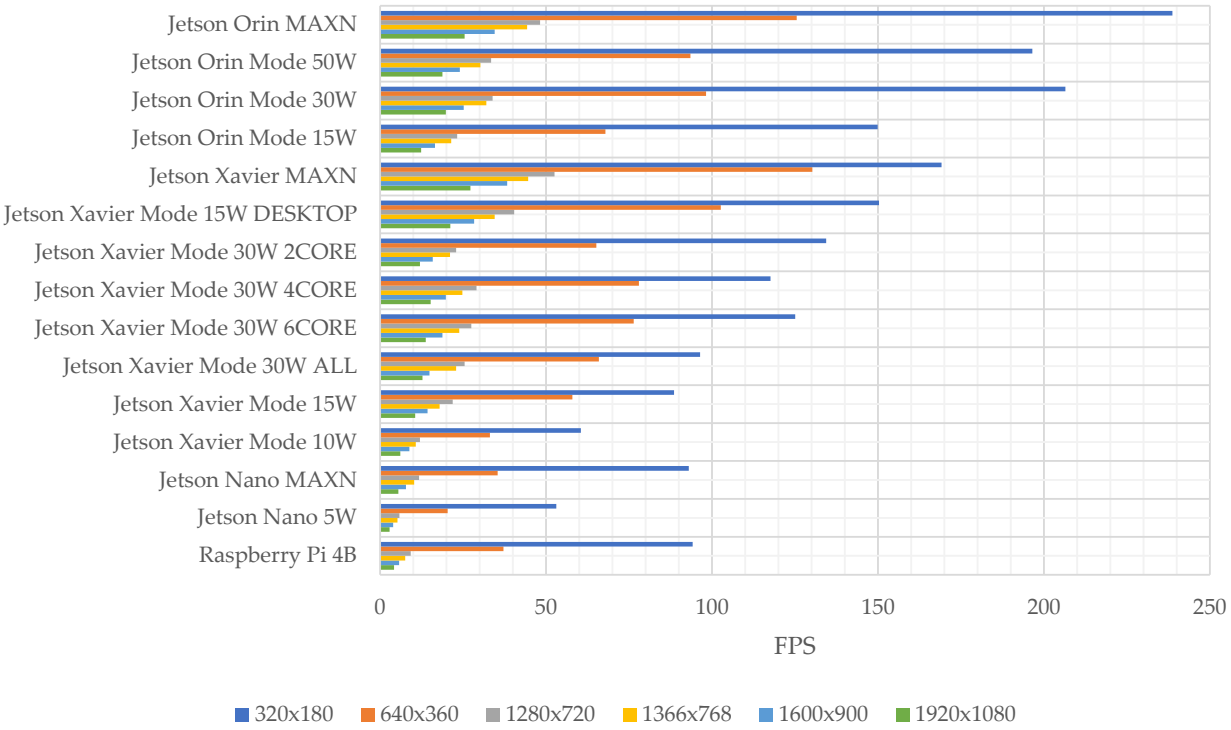


Figure 7. Graph of FPS values of image segmentation in the HSV color space algorithm for various power modes of tested platforms.

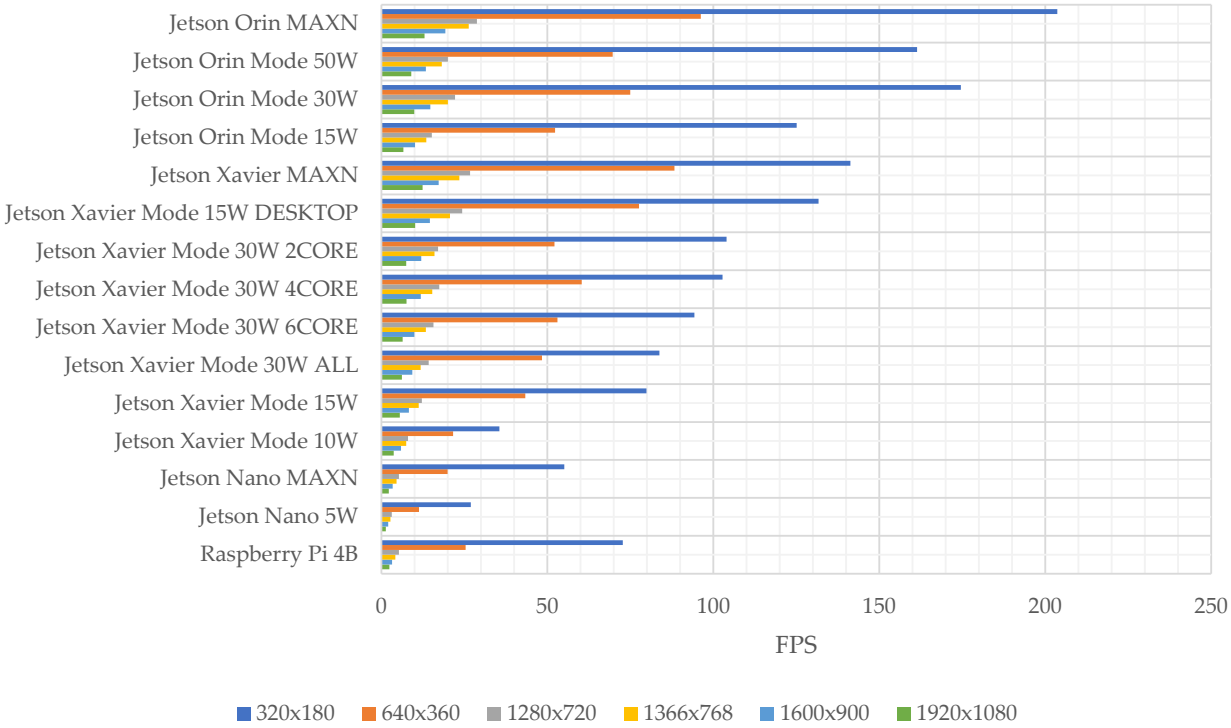


Figure 8. FPS of line detection using the Hough algorithm with Scharr mask filtering for various power modes of tested platforms.

Due to the architecture of the airport line detection program, which is single-threaded [5], reducing the number of active processor cores does not have a negative impact on the achieved results. The computational complexity of line detection using the Hough algorithm with Scharr mask filtering demonstrates the increase in input processing time in

embedded systems. For this reason, performance at the level of 24 FPS of video sequence processing was possible only after reducing the resolution to 1366 × 768, where the values of 24 FPS were obtained for the most efficient NVIDIA Jetson Xavier MAXN modules and 26 FPS for NVIDIA Jetson Orin MAXN.

The efficiency of line detection using the Hough algorithm with Scharr mask filtering, due to the greater computational complexity, causes the maximum performance of the tested embedded devices to be worse than in the case of the algorithm based on image segmentation in the HSV color space, making the Hough algorithm unequally usable in real-time work.

4.2. Power Consumption

4.2.1. Software Power Consumption Measurement

Energy consumption was measured using the Jetson-Stats software [30]. It is an advanced system monitoring and control package explicitly designed for the NVIDIA Jetson series, including the Orin, Xavier, Nano, and TX models. The tool (Figure 9) is a vital resource for researchers and developers seeking in-depth analysis and performance tracking of their NVIDIA Jetson boards. Key features of Jetson-Stats include:

- Hardware, architecture, L4T, and NVIDIA Jetpack decoding: The tool provides detailed information about the system's underlying hardware and software configurations, thereby offering insights into the system's operational parameters and potential performance-optimization opportunities.
- Comprehensive monitoring: Jetson-Stats can track a wide range of system metrics, including CPU, GPU, memory, engines, and fan speeds. This monitoring feature ensures continuous awareness of system health and performance.
- System control: The package allows users to manage their NVP model, fan speed, and jetson_clocks, offering comprehensive control over the performance and power management of the device.

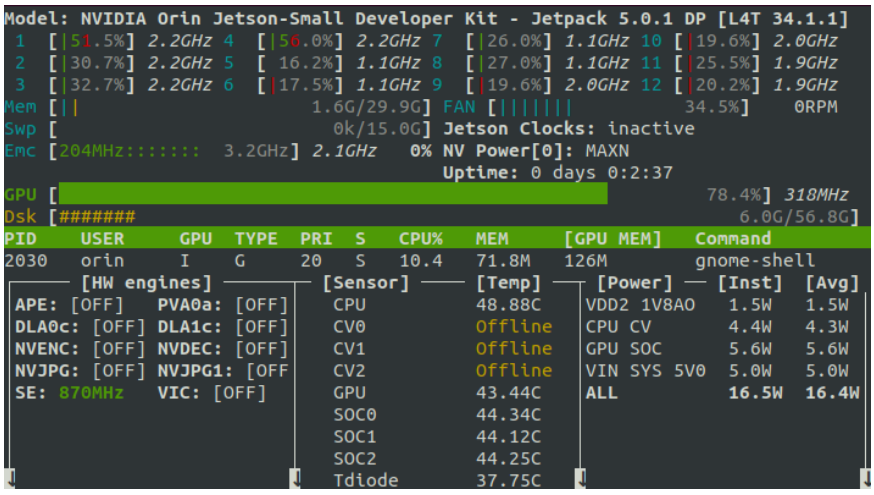


Figure 9. Jetson-Stats software interface running on NVIDIA Jetson Orin AGX.

The resolution of the average value of power consumption is shown at the level of 0.1 W. Due to the compatibility of the results obtained using the Jetson-Stats software, the results for the hardware measurement using multimeters are also presented with the same accuracy. Table 5 presents the results of measurements of the average power consumption during the operation of the algorithm for various power modes and different resolutions using the algorithm based on image segmentation in the HSV color space. Table 6 presents the results for line detection using the Hough algorithm with Scharr mask filtering. The tables show the average results of the offline trials for the 10 s sequences mentioned above,

with a change in resolution, randomly selected from the database pool. Measurements were made with no FPS limits.

Table 5. Average power consumption (in Watts) for the algorithm based on image segmentation in the HSV color space using Jetson-Stats software.

	320 × 180	640 × 360	1280 × 720	1366 × 768	1600 × 900	1920 × 1080
Jetson Nano 5 W	2.8	3.2	3.1	3.2	3	3.2
Jetson Nano MAXN	4.8	4.9	4.8	5	5	4.8
Jetson Xavier Mode 10 W	4.3	4.3	4.3	4.1	4.1	4.1
Jetson Xavier Mode 15 W	4.6	4.9	4.9	4.9	5	4.9
Jetson Xavier Mode 30 W ALL	5	5.5	5.4	5.4	5.4	5.4
Jetson Xavier Mode 30 W 6CORE	5.4	5.5	5.6	5.7	5.8	6.1
Jetson Xavier Mode 30 W 4CORE	5.5	5.9	6.2	6.2	6.5	6.5
Jetson Xavier Mode 30 W 2CORE	6.8	7.1	7.3	7.3	7.3	7.4
Jetson Xavier Mode 15 W DESKTOP	7.5	8.7	8.9	9.1	9.2	9.4
Jetson Xavier MAXN	10.7	13.2	13.1	13.2	13.4	13.5
Jetson Orin Mode 15 W	11.3	11.4	11.5	11.5	11.5	11.7
Jetson Orin Mode 30 W	12.5	13.4	13.3	13.3	13.3	13.9
Jetson Orin Mode 50 W	12.1	13.1	12.5	12.5	13	13.9
Jetson Orin MAXN	15.4	17.7	16.7	16.8	16.8	16.9

Table 6. Average power consumption (in Watts) for line detection using the Hough algorithm with Scharr mask filtering using Jetson-Stats software.

	320 × 180	640 × 360	1280 × 720	1366 × 768	1600 × 900	1920 × 1080
Jetson Nano 5 W	3.1	3	3	3	3	3.1
Jetson Nano MAXN	4.1	3.9	4.4	4.7	4.6	4.8
Jetson Xavier Mode 10 W	4.1	4.1	4.1	4.1	4.1	4.1
Jetson Xavier Mode 15 W	4.6	5.1	4.7	4.6	4.6	4.6
Jetson Xavier Mode 30 W ALL	4.9	5.2	4.9	4.9	4.9	4.7
Jetson Xavier Mode 30 W 6CORE	5	5.4	5.2	5.2	5.2	5.1
Jetson Xavier Mode 30 W 4CORE	5.3	5.9	5.7	5.7	5.9	5.5
Jetson Xavier Mode 30 W 2CORE	6.7	6.9	6.9	7.1	7.1	7.1
Jetson Xavier Mode 15 W DESKTOP	7.4	8.1	8	7.9	8.2	7.9
Jetson Xavier MAXN	10.3	11.5	11.2	11.1	10.9	10.9
Jetson Orin Mode 15 W	11.2	11.3	11.4	11.4	11.4	11.3
Jetson Orin Mode 30 W	12.7	12.9	12.7	12.7	12.7	12.7
Jetson Orin Mode 50 W	12.1	12.5	11.9	11.9	11.9	11.9
Jetson Orin MAXN	15.4	16.7	15.6	15.3	15.3	14.9

More advanced modules, e.g., NVIDIA Jetson Orin AGX working in MAXN mode and NVIDIA Jetson Xavier AGX in MAXN mode, have higher power requirements, which is quite an obvious observation, because at the same time, the FPS obtained for these devices is definitely higher, as shown in Section 4.1. For individual power modes, there is a slight variation in power consumption at different resolutions, and the measured values are quite similar. This is because there are no limits to the FPS values, which were individually

maximum for each of the experiments. To better assess the power requirements, which depend on both the type of algorithm, the resolution used, and the FPS value obtained, these dependencies have been taken into account and summarized in the next sub-chapter.

4.2.2. Power Consumption Measurement Using an Electronic Multimeter

Due to the use of an embedded system from the Raspberry Pi family in the paper, it was not possible to use the above-mentioned software. For this reason, the authors decided to use standard laboratory multimeters. In this way, the average energy consumption of a given embedded system was obtained. It is worth noting that the earlier Jetson-Stats software solution does not take into account the power consumption of accessories connected to the microcomputer, but only measures the power consumed by its components. In the following cases, measurements include typical accessories connected to microcomputers, in the form of a keyboard, mouse, and a fan.

Table 7 and Figure 10 presents the results of the average power consumption measurements during the operation of the algorithm for various power modes and different resolutions using the algorithm based on image segmentation in the HSV color space with the use of a power meter. Table 8 and Figure 11 present the results for line detection using the Hough algorithm with Scharr mask filtering.

For the algorithm analyzed based on image segmentation in the HSV color space, the highest resolution of 1920 × 1080 implies a limit to the FPS value below 24, which is an acceptable value for airport line detection techniques. These assumptions are met by two microcomputers operating in the following configurations: NVIDIA Jetson Xavier AGX MAXN and NVIDIA Jetson Orin AGX in mode MAXN. In these cases, the average power consumption is:

- For NVIDIA Jetson Xavier MAXN: 18.5 W @ 27 FPS
- For NVIDIA Jetson Orin MAXN: 20.8 W @ 25 FPS

Table 7. Average power consumption (in Watts) for the algorithm based on image segmentation in the HSV color space using electronic multimeter.

	320 × 180	640 × 360	1280 × 720	1366 × 768	1600 × 900	1920 × 1080
Raspberry Pi 4B	5.2	5.7	5.4	5.4	5.4	5.4
Jetson Nano 5 W	4.2	4.2	4.4	4.4	4.4	4.3
Jetson Nano MAXN	5.5	6.6	5.5	6.4	6.0	5.8
Jetson Xavier Mode 10 W	8.3	8.3	8.4	8.3	8.4	8.5
Jetson Xavier Mode 15 W	8.9	9.1	9.3	9.2	9.3	9.3
Jetson Xavier Mode 30 W ALL	9.4	9.8	9.8	9.8	10.0	10.0
Jetson Xavier Mode 30 W 6CORE	9.7	10.0	10.2	10.2	10.1	10.3
Jetson Xavier Mode 30 W 4CORE	9.8	10.3	10.7	10.6	10.7	10.8
Jetson Xavier Mode 30 W 2CORE	11.3	11.6	11.8	11.8	12.0	12.0
Jetson Xavier Mode 15 W DESKTOP	12.1	13.1	13.7	13.7	13.9	13.9
Jetson Xavier MAXN	15.3	18.3	18.4	18.0	18.6	18.5
Jetson Orin Mode 15 W	15.0	15.2	15.0	15.4	15.5	15.7
Jetson Orin Mode 30 W	15.6	16.5	17.2	17.1	17.2	18.0
Jetson Orin Mode 50 W	16.2	16.5	16.7	16.4	16.7	17.6
Jetson Orin MAXN	20.1	21.9	20.4	20.7	20.8	20.8

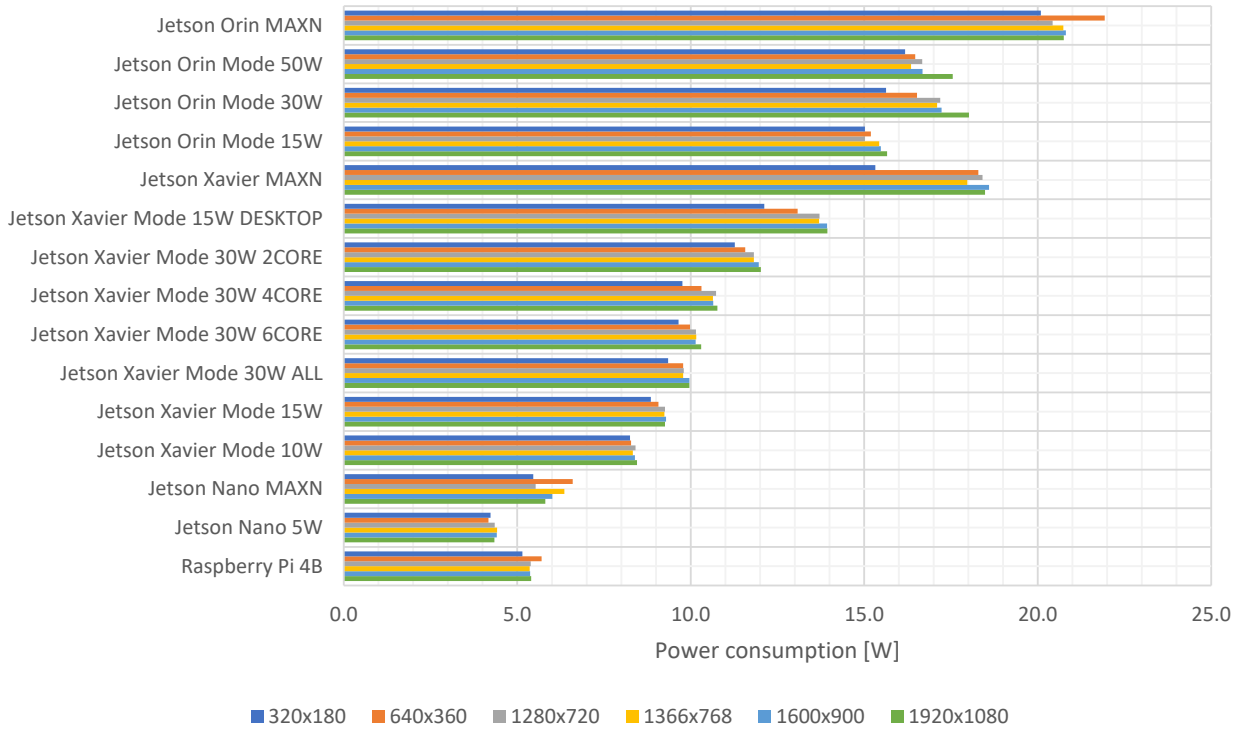


Figure 10. Average power consumption (in Watts) for algorithm based on image segmentation in the HSV color space using an electronic multimeter.

Table 8. Average power consumption (in Watts) for line detection using the Hough algorithm with Scharr mask filtering using an electronic multimeter.

	320 × 180	640 × 360	1280 × 720	1366 × 768	1600 × 900	1920 × 1080
Raspberry Pi 4B	5.2	5.4	5.2	5.1	5.2	5.7
Jetson Nano 5 W	4.6	4.4	4.5	4.2	4.2	4.3
Jetson Nano MAXN	5.3	5.2	5.3	5.3	5.2	5.3
Jetson Xavier Mode 10 W	8.2	8.2	8.3	8.4	8.4	8.4
Jetson Xavier Mode 15 W	8.8	8.9	8.9	8.8	8.8	8.8
Jetson Xavier Mode 30 W ALL	9.1	9.5	9.1	9.1	9.0	8.9
Jetson Xavier Mode 30 W 6CORE	9.3	9.6	9.6	9.6	9.4	9.4
Jetson Xavier Mode 30 W 4CORE	9.7	10.1	10.1	10.1	10.2	9.8
Jetson Xavier Mode 30 W 2CORE	11.1	11.4	11.6	11.6	11.6	11.6
Jetson Xavier Mode 15 W DESKTOP	11.8	12.7	12.9	12.9	12.8	12.3
Jetson Xavier MAXN	15.3	16.4	15.9	16.0	15.6	15.8
Jetson Orin Mode 15 W	15.1	15.1	15.2	15.0	15.2	15.1
Jetson Orin Mode 30 W	16.4	16.6	16.4	17.0	16.7	16.3
Jetson Orin Mode 50 W	15.1	16.1	16.0	15.9	16.0	16.0
Jetson Orin MAXN	18.5	20.3	20.3	20.0	19.4	19.4

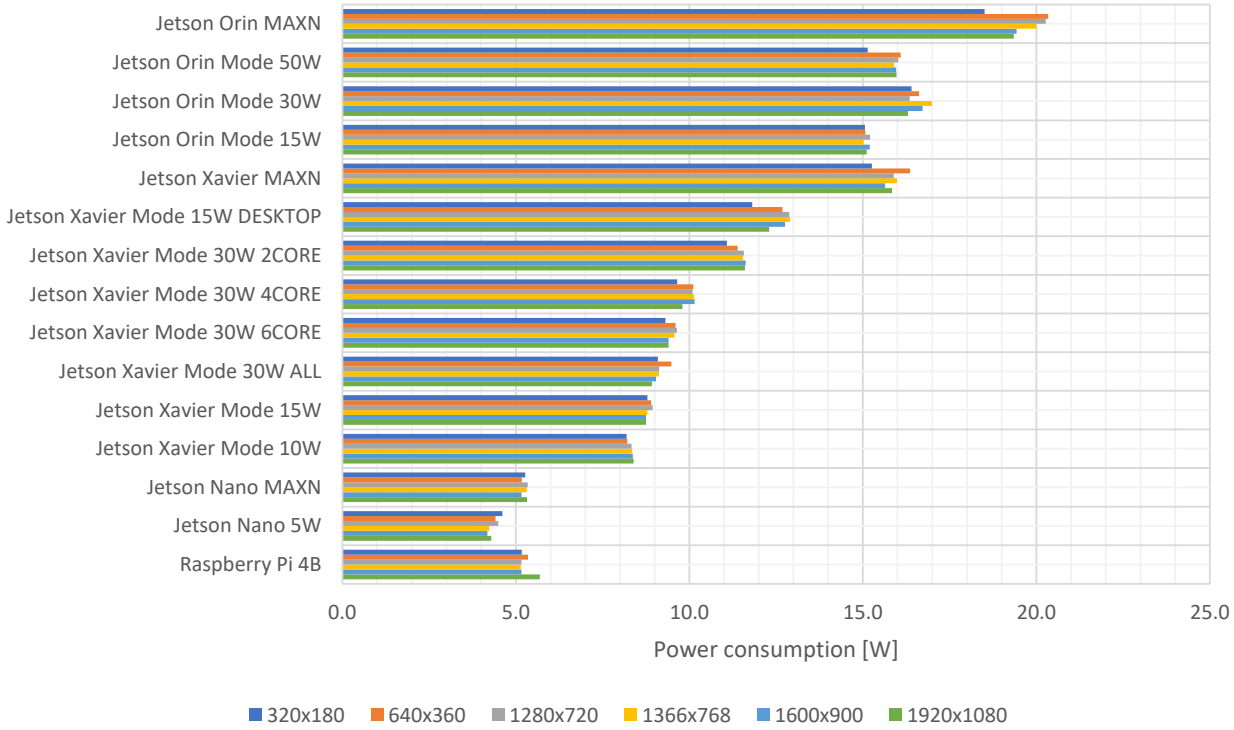


Figure 11. Average power consumption (in Watts) for line detection using the Hough algorithm with Scharr mask filtering using an electronic multimeter.

In the case of the second solution, line detection used the Hough algorithm with Scharr mask filtering and, as before, there was a limit set for processing at a minimum speed of 24 FPS, with a resolution of video sequences of 1920 × 1080. For this case, none of the microcomputers reached the minimum FPS value. Only the resolution of 1366 × 768 allowed processing of video sequences with sufficient speed. These assumptions are also met by two microcomputers operating in the following configurations: NVIDIA Jetson Xavier AGX in MAXN mode and NVIDIA Jetson Orin AGX in MAXN mode. In these cases, the average power consumption is:

- For NVIDIA Jetson Xavier MAXN: 16 W @ 24 FPS
- For NVIDIA Jetson Orin MAXN: 20 W @ 26 FPS

4.3. Energy Efficiency Analysis

For the FPS and power consumption values determined in Sections 4.1 and 4.2, respectively, comparative charts have been prepared that show the dependence of power consumption requirements on both the resolution and the FPS values. Figure 12 shows the dependence of the algorithm based on image segmentation in the HSV color space. The lower the value, the lower the power requirements. For faster processing speeds expressed in FPS, the MAX modes are the best.

The comparison of the operation of the two algorithms shows that in the case of an algorithm based on image segmentation in the HSV color space, it is possible to obtain the same or better FPS values, with less energy consumption. For example, for the NVIDIA Jetson Xavier in mode MAXN device, this algorithm for 1920 × 1080 resolution requires 679 mJ/frame, while line detection using the Hough algorithm with Scharr mask filtering required 680 mJ/frame for 1366 × 768 resolution.

Figure 13 shows the dependence graph for line detection using the Hough algorithm with Scharr mask filtering.

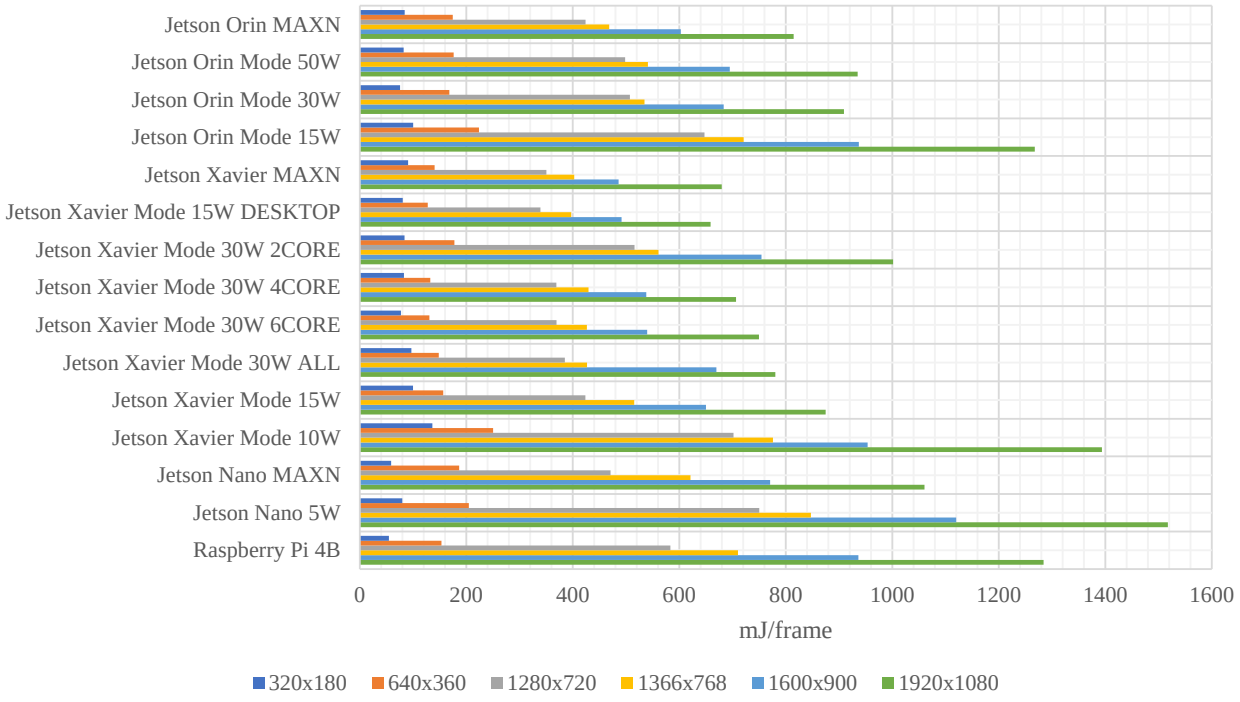


Figure 12. Energy efficiency analysis of the algorithm based on image segmentation in the HSV color space.

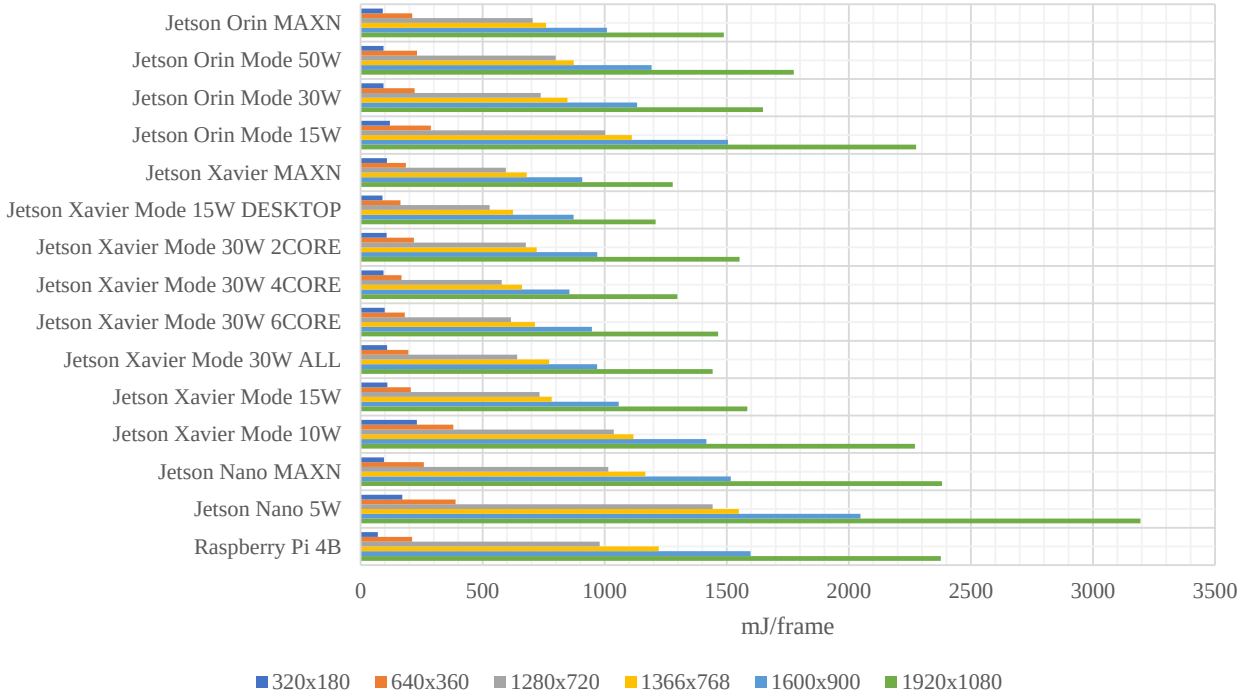


Figure 13. Energy efficiency analysis of line detection using the Hough algorithm with Scharr mask filtering.

4.4. Dynamic Voltage and Frequency Switching (DVFS)

There is also another approach to control hardware performance similar to NVIDIA Jetson manufacturer power modes—dynamic voltage and frequency switching (DVFS) [31–33]. As part of the tests, an experiment was also carried out to run the algorithms image segmentation in the HSV color space and line detection using the Hough algorithm with Scharr mask filtering, as was the case with previous research. Energy consumption was also measured using multimeters

and Jetson-Stats software. This time, however, not only the power mode of the microcomputer was changed, but also the DVFS policy (possible modes: schedutil, performance, powersave, userspace, on-demand, interactive, conservative). The tests were carried out on the NVIDIA Jetson AGX Xavier microcomputer, due to the best results achieved in previous tests.

Concerning the assumptions about processing speed, the NVIDIA Jetson AGX Xavier microcomputer also only met values in the MAXN mode in DVFS policies: schedutil (27 fps), performance (25 fps), on-demand (27 fps), interactive (26 fps), and conservative (27 fps) for the image segmentation in the HSV color space for a resolution of 1920 × 1080. In the case of the line detection algorithm using the Hough algorithm with Scharr mask filtering, the assumptions were not achieved. Figures 14 and 15 show the exact results of the experiment. It is worth mentioning that the result in the MAXN and DVFS policy powersave operating modes resulted in a practical inability to run the program, which in turn significantly extended the program's running time, and thus the energy consumption per image frame.

The experiments carried out, as part of the assumptions and algorithms used, show that the best operating modes that meet the assumptions are powermode MAXN and DVFS policy schedutil (680 mJ/frame); however, in most cases, the differences are in the range of 7%:

- schedutil: 27 fps 680 mJ/frame
- performance: 25 fps 732 mJ/frame
- on-demand: 27 fps 697 mJ/frame
- interactive: 26 fps 698 mJ/frame
- conservative 27 fps 685 mJ/frame

It is worth noting that the previously obtained results (default DVFS policy schedutil) confirm the optimization of the mode, which among all tested ones gave the best result (lowest energy consumption per image frame).

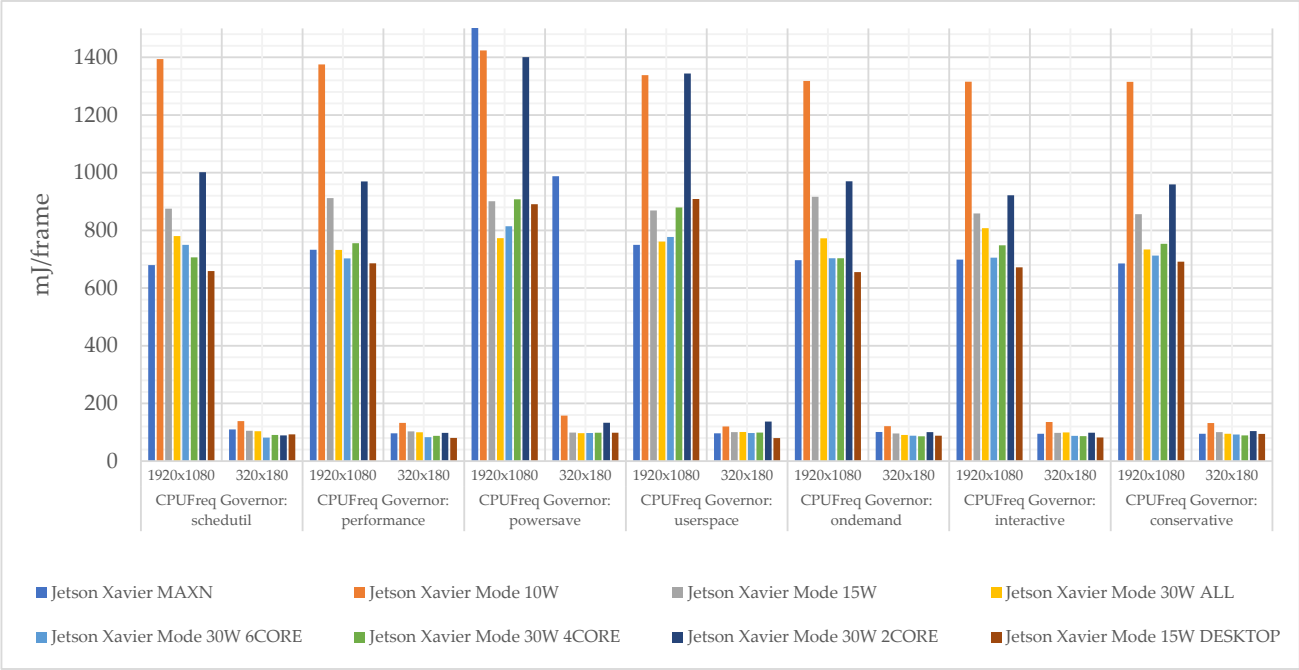


Figure 14. Energy efficiency analysis of the algorithm based on image segmentation in the HSV color space using different DVFS policies.

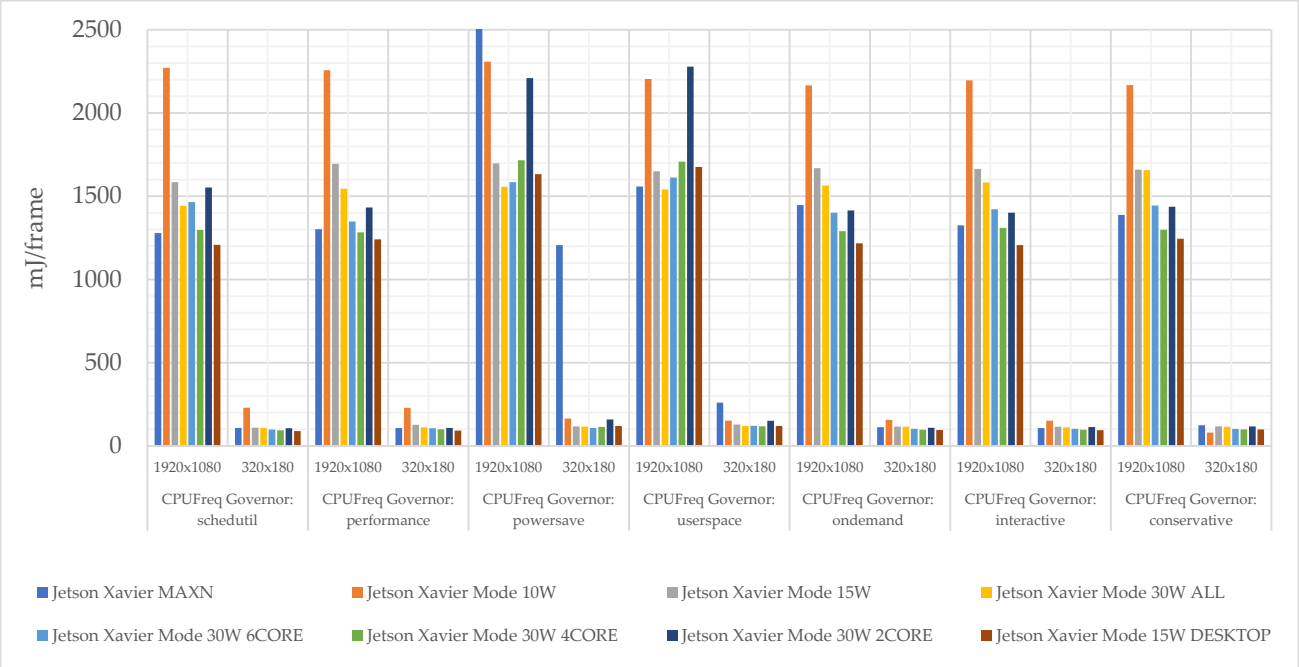


Figure 15. Energy efficiency analysis of line detection using the Hough algorithm with Scharr mask filtering using different DVFS policies.

5. Conclusions and Future Work

New development modules using advanced microprocessors, and at the same time not having high power requirements (the ability to operate for several hours using battery power), can be successfully used in mobile monitoring devices.

The simplest Raspberry Pi 4B modules are suitable for solutions where resolution requirements are not too high. As a series of experiments has shown, standard values at the 30 FPS level are processed with the use of such modules at a maximum resolution of 640 × 360. For such resolutions, the advantage of using Raspberry Pi 4B modules is the low cost of the device and relatively low energy demand to process one frame of the video sequence. This device can work with passive cooling, and it is not necessary to use an additional fan, which required an additional 0.5 W during the tests. An alternative solution to using the Raspberry Pi 4B is to use the NVIDIA Jetson Nano module. This device has two operating modes: 5 W and MAXN. In MAXN mode, it has performance and energy requirements similar to those of Raspberry Pi 4B.

The most efficient units in the form of NVIDIA Jetson Xavier AGX and NVIDIA Jetson Orin AGX are the only modules that meet the assumptions of the FPS value when processing video sequences. Despite the fact that the latest solution of the manufacturer is NVIDIA Jetson Orin AGX, in the vast majority of tests carried out in the article, NVIDIA Jetson Xavier AGX achieved better results, not only in FPS values but also in the ratio of the necessary power to process one frame, especially in default schedutil DVFS policy.

The specifics of the evaluation and selection of embedded system devices is a complex issue due to several different aspects discussed in this article. The energy aspects, important from the point of view of mobile devices, have been noticed by the manufacturers of experimental modules, who in the offered solutions provide the possibility of using various power modes. At the same time, it can be noted that there are currently few publications that comprehensively assess the use of such opportunities.

It should be noted that the examined algorithms have a typical scale of difficulty in processing video sequences. Thus, the estimates presented in the paper can be valuable guidelines for designers of intelligent embedded systems processing video sequences. Of course, in order to precisely determine energy requirements, selected algorithms should be run individually.

The assessment presented in this paper is related to the authors' previous practical solutions, while the tested sets of image processing blocks are quite universal in most video sequence processing solutions.

Author Contributions: Conceptualization, J.S., K.P. and T.M.; methodology, J.S., K.P. and T.M.; software, J.S. and K.P.; validation, J.S., K.P. and T.M.; formal analysis, J.S., K.P. and T.M.; investigation, J.S., K.P. and T.M.; resources, J.S. and K.P.; writing—original draft preparation, J.S., K.P. and T.M.; writing—review and editing, J.S., K.P. and T.M.; visualization, J.S. and K.P.; supervision, J.S., K.P. and T.M.; project administration, J.S., K.P. and T.M.; funding acquisition, J.S., K.P. and T.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded partly by the 2023 subvention and partly with the SMART4ALL EU Horizon 2020 project, Grant Agreement No 872614.

Data Availability Statement: The software used in this paper are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Arshad, N.M.; Razak, N.A. Vision-based detection technique for effective line-tracking autonomus vehicle. In Proceedings of the 2012 IEEE 8th International Colloquium on Signal Processing and its Applications, Malacca, Malaysia, 23–25 March 2012; pp. 441–445. [CrossRef]
2. Barua, B.; Gomes, C.; Baghe, S.; Sisodia, J. Self-Driving Car Implementation using Computer Vision for Detection and Navigation. In Proceedings of the 2019 International Conference on Intelligent Computing and Control Systems (ICCS), Madurai, India, 15–17 May 2019; pp. 271–274. [CrossRef]
3. Elmanaa, I.; Sabri, M.A.; Abouch, Y.; Aarab, A. Efficient Roundabout Supervision: Real-Time Vehicle Detection and Tracking on Nvidia Jetson Nano. *Appl. Sci.* **2023**, *13*, 7416. [CrossRef]
4. Suder, J.; Maciejewski, P.; Podbucki, K.; Marciniak, T.; Dąbrowski, A. Measuring Platform for Quality Testing of Airport Lamps. *Pomiary Autom. Robot.* **2019**, *23*, 5–13. [CrossRef]
5. Suder, J.; Podbucki, K.; Marciniak, T.; Dąbrowski, A. Low Complexity Lane Detection Methods for Light Photometry System. *Electronics* **2021**, *10*, 1665. [CrossRef]
6. European Union Aviation Safety Agency. Certification Specifications and Guidance Material for Aerodrome Design (CS-ADR-DSN). 29 March 2022. Available online: https://www.easa.europa.eu/en/downloads/136283/en (accessed on 30 June 2022).
7. Jabłoński, B.; Makowski, D.; Perek, P.; Nowak vel Nowakowski, P.; Sitjes, A.P.; Jakubowski, M.; Gao, Y.; Winter, A.; WX Team. Evaluation of NVIDIA Xavier NX Platform for Real-Time Image Processing for Plasma Diagnostics. *Energies* **2022**, *15*, 2088. [CrossRef]
8. Jabłoński, B.; Makowski, D.; Perek, P. Implementation of Thermal Event Image Processing Algorithms on NVIDIA Tegra Jetson TX2 Embedded System-on-a-Chip. *Energies* **2021**, *14*, 4416. [CrossRef]
9. Suder, J. Parameters evaluation of cameras in embedded systems. *Przegląd Elektrotechniczny* **2022**, *98*, 218–221. [CrossRef]
10. Podbucki, K.; Suder, J.; Marciniak, T.; Dabrowski, A. Evaluation of Embedded Devices for Real- Time Video Lane Detection. In Proceedings of the 2022 29th International Conference on Mixed Design of Integrated Circuits and System (MIXDES), Wrocław, Poland, 23–24 June 2022; pp. 187–191. [CrossRef]
11. Revolution Pi. Available online: https://revolutionpi.com/revolution-pi-series (accessed on 10 July 2023).
12. CPU Benchmarks, PassMark®Software. Available online: https://www.cpubenchmark.net/cpu.php?cpu=BCM2711&id=4297 (accessed on 10 July 2023).
13. Jetson Benchmarks, Nvidia Developer. Available online: https://developer.nvidia.com/embedded/jetson-benchmarks (accessed on 5 July 2023).
14. Yildirim, M.; Karaduman, O.; Kurum, H. Real-Time Image and Video Processing Applications Using Raspberry Pi. In Proceedings of the 2022 IEEE 1st Industrial Electronics Society Annual On-Line Conference (ONCON), Kharagpur, India, 9–11 December 2022; pp. 1–6. [CrossRef]
15. Lee, S.J. Challenges of Real-time Processing with Embedded Vision for IoT Applications. In Proceedings of the 2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), Maldives, Maldives, 16–18 November 2022; pp. 1–6. [CrossRef]
16. Annusewicz, A.; Zwierzchowski, J. Marker Detection Algorithm for the Navigation of a Mobile Robot. In Proceedings of the 2020 27th International Conference on Mixed Design of Integrated Circuits and System (MIXDES), Lodz, Poland, 25–27 June 2020; pp. 223–226. [CrossRef]
17. Barnell, M.; Raymond, C.; Smiley, S.; Isereau, D.; Brown, D. Ultra Low-Power Deep Learning Applications at the Edge with Jetson Orin AGX Hardware. In Proceedings of the 2022 IEEE High Performance Extreme Computing Conference (HPEC), Waltham, MA, USA, 19–23 September 2022; pp. 1–4. [CrossRef]

18. Kortli, Y.; Gabsi, S.; Voon, L.F.C.L.Y.; Jridi, M.; Merzougui, M.; Atri, M. Deep embedded hybrid CNN–LSTM network for lane detection on NVIDIA Jetson Xavier NX. *Knowl. Based Syst.* **2022**, *240*, 107941. [CrossRef]
19. Choe, C.; Choe, M.; Jung, S. Run Your 3D Object Detector on NVIDIA Jetson Platforms: A Benchmark Analysis. *Sensors* **2023**, *23*, 4005. [CrossRef] [PubMed]
20. Suzen, A.A.; Duman, B.; Sen, B. Benchmark Analysis of Jetson TX2, Jetson Nano and Raspberry PI using Deep-CNN. In Proceedings of the 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Ankara, Turkey, 26–28 June 2020; pp. 1–5. [CrossRef]
21. Vijitkunsawat, W.; Chantngarm, P. Comparison of Machine Learning Algorithms on Self-Driving Car Navigation using Nvidia Jetson Nano. In Proceedings of the 2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Phuket, Thailand, 24–27 June 2020; pp. 201–204. [CrossRef]
22. Jain, A.K. Working model of Self-driving car using Convolutional Neural Network, Raspberry Pi and Arduino. In Proceedings of the 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 29–31 March 2018; pp. 1630–1635. [CrossRef]
23. Civik, E.; Yuzgec, U. Real-time driver fatigue detection system with deep learning on a low-cost embedded system. *Microprocess. Microsyst.* **2023**, *99*, 104851. [CrossRef]
24. Raspberry Pi 4, 3A+, Zero W—Specs, Benchmarks & Thermal Tests. Available online: https://magpi.raspberrypi.com/articles/raspberry-pi-specs-benchmarks (accessed on 10 June 2023).
25. NVIDIA CORPORATION. NVIDIA Jetson Linux Developer Guide 32.7.1 Release. Available online: https://docs.nvidia.com/jetson/l4t/index.html#page/Tegra%20Linux%20Driver%20Package%20Development%20Guide/power_management_nano.html (accessed on 5 June 2023).
26. Jetson Orin NX Series and Jetson AGX Orin Series. Available online: https://docs.nvidia.com/jetson/archives/r35.1/DeveloperGuide/text/SD/PlatformPowerAndPerformance/JetsonOrinNxSeriesAndJetsonAgxOrinSeries.html (accessed on 10 July 2023).
27. GoPro. GoPro Hero+ LCD User Manual. Available online: https://gopro.com/content/dam/help/heroplus-lcd/manuals/UM_HEROPlusLCD_ENG_REVB_WEB.pdf (accessed on 5 June 2023).
28. Park, M.S.; Brock, A.; Mortimer, V.; Taussky, P.; Couldwell, W.T.; Quigley, E. GoPro Hero Cameras for Creation of a Three-Dimensional, Educational, Neurointerventional Video. *J. Digit. Imaging* **2017**, *30*, 561–565. [CrossRef] [PubMed]
29. Roy, A.; Biswas, N. GoPro: A Low Complexity Task Allocation Algorithm for a Mobile Edge Computing System. In Proceedings of the 2022 National Conference on Communications (NCC), Mumbai, India, 24–27 May 2022; pp. 413–418. [CrossRef]
30. Simple Package for Monitoring and Control Your NVIDIA Jetson [Orin, Xavier, Nano, TX] Series. Available online: https://github.com/rbonghi/jetson_stats (accessed on 5 May 2023).
31. Chen, Y.-L.; Chang, M.-F.; Yu, C.-W.; Chen, X.-Z.; Liang, W.-Y. Learning-Directed Dynamic Voltage and Frequency Scaling Scheme with Adjustable Performance for Single-Core and Multi-Core Embedded and Mobile Systems. *Sensors* **2018**, *18*, 3068. [CrossRef] [PubMed]
32. Kang, D.-K.; Lee, K.-B.; Kim, Y.-C. Cost Efficient GPU Cluster Management for Training and Inference of Deep Learning. *Energies* **2022**, *15*, 474. [CrossRef]
33. Khriji, S.; Chéour, R.; Kanoun, O. Dynamic Voltage and Frequency Scaling and Duty-Cycling for Ultra Low-Power Wireless Sensor Nodes. *Electronics* **2022**, *11*, 4071. [CrossRef]