Preface

Tetiana A. Vakaliuk[1,2,3,4], Serhiy O. Semerikov[3,2,5,1,4]

[1]Zhytomyr Polytechnic State University, 103 Chudnivsyka Str., Zhytomyr, 10005, Ukraine
[2]Institute for Digitalisation of Education of the NAES of Ukraine, 9 M. Berlynskoho Str., Kyiv, 04060, Ukraine
[3]Kryvyi Rih State Pedagogical University, 54 Universytetskyi Ave., Kryvyi Rih, 50086, Ukraine
[4]Academy of Cognitive and Natural Sciences, 54 Universytetskyi Ave., Kryvyi Rih, 50086, Ukraine
[5]Kryvyi Rih National University, 11 Vitalii Matusevych Str., Kryvyi Rih, 50027, Ukraine

Abstract

This is an introduction to a collection of selected papers from the Edge Computing Workshop (doors 2024), held in Zhytomyr, Ukraine, on April 05, 2024. The workshop covers topics such as algorithms and techniques for machine learning and AI at the edge, cellular infrastructure for edge computing, distributed ledger technology and blockchain at the edge, edge computing infrastructure and edge-enabled applications, edge-based data storage and databases, edge-optimized heterogeneous architectures, fault-tolerance in edge computing, fog computing models and applications, geo-distributed analytics and indexing on edge nodes, hardware architectures for edge computing and devices, innovative applications at the edge, interoperability and collaboration between edge and cloud computing, monitoring, management, and diagnosis in edge computing, processing of IoT data at network edges, programming models and toolkits for edge computing, resource management and Quality of Service for edge computing, security and privacy in edge computing and others. The workshop proceedings consist of an introduction and nine accepted articles that were painstakingly modified by the authors based on the discussion outcomes and were presented by the authors at the workshop. The papers were rigorously peer-reviewed and selected from 19 submissions.

Keywords

edge computing, edge device, IoT, UAV, distributed systems

1. Introduction

Edge Computing Workshop (doors) is a peer-reviewed international Computer Science workshop focusing on research advances and applications of edge computing, a process of building a distributed system in which some applications, as well as computation and storage services, are provided and managed by central clouds and smart devices, the edge of networks in small proximity to mobile devices, sensors, and end users; and others are provided and managed by the center cloud and a set of small in-between local clouds supporting IoT at the edge.

Since 2021, the workshop covers topics such as algorithms and techniques for machine learning and AI at the edge, cellular infrastructure for edge computing, distributed ledger technology and blockchain at the edge, edge computing infrastructure and edge-enabled applications, edge-based data storage and databases, edge-optimized heterogeneous architectures, fault-tolerance in edge computing, fog computing models and applications, geo-distributed analytics and indexing on edge nodes, hardware architectures for edge computing and devices, innovative applications at the edge, interoperability and collaboration between edge and cloud computing, monitoring, management, and diagnosis in edge computing, processing of IoT data at network edges, programming models and toolkits for edge computing, resource management and Quality of Service for edge computing, security and privacy in edge computing and others.

This volume represents the proceedings of the 4th Edge Computing Workshop (doors 2024), held in Zhytomyr, Ukraine, on April 05, 2024. It comprises 9 contributed articles that have been carefully peer-reviewed and selected from 19 submissions. At least three program committee members have

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine

tetianavakaliuk@acnsci.org (T. A. Vakaliuk); semerikov@ccjournals.eu (S. O. Semerikov)
http://acnsci.org/vakaliuk/ (T. A. Vakaliuk); https://kdpu.edu.ua/semerikov (S. O. Semerikov)
0000-0001-6825-4697 (T. A. Vakaliuk); 0000-0003-0789-0272 (S. O. Semerikov)

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

examined each contribution, and they have all been reviewed for plagiarism, self-plagiarism, and fair referencess.

2. doors 2024 committees

2.1. Program committee co-chairs

- *Tetiana A. Vakaliuk*, Zhytomyr State Polytechnic University, Ukraine
- *Serhiy O. Semerikov*, Kryvyi Rih State Pedagogical University, Ukraine

2.2. Program committee

- *Aleksandr Cariow*, West Pomeranian University of Technology, Poland
- *Attila Kertesz*, University of Szeged, Hungary
- *Nagender Kumar Suryadevara*, University of Hyderabad, India
- *Gyu Myoung Lee*, Liverpool John Moores University, United Kingdom
- *BongKyo Moon*, Dongguk University, South Korea
- *Michael J. O'Grady*, University College Dublin, Ireland
- *Pedro Valderas*, Universitat Politècnica de València, Spain
- *Xianzhi Wang*, University of Technology Sydney, Australia
- *Eiko Yoneki*, University of Cambridge, United Kingdom
- *Alejandro Zunino*, ISISTAN Research Institute, UNCPBA & CONICET, Argentina

2.3. Additional reviewers

- *Olexander Barmak*, Khmelnytskyi National University, Ukraine
- *Akinul Islam Jony*, American International University-Bangladesh, Bangladesh
- *Valerii Kontsedailo*, Inner Circle, Netherlands
- *Vyacheslav Kryzhanivskyy*, R&D Seco Tools AB, Sweden
- *Nadiia Lobanchykova*, Zhytomyr Polytechnic State University, Ukraine
- *Mykhailo Medvediev*, ADA University, Azerbaijan
- *Franco Milano*, University of Florence, Italy
- *Tetiana Nikitchuk*, Zhytomyr Polytechnic State University, Ukraine
- *Etibar Seyidzade*, Baku Engineering University, Azerbaijan
- *Andrii Striuk*, Kryvyi Rih National University, Ukraine

2.4. Organizing committee

- *Tetiana Nikitchuk*, Zhytomyr Polytechnic State University, Ukraine
- *Andrii Morozov*, Zhytomyr Polytechnic State University, Ukraine
- *Serhiy Semerikov*, Kryvyi Rih State Pedagogical University, Ukraine
- *Andrii Striuk*, Kryvyi Rih National University, Ukraine
- *Tetiana Vakaliuk*, Zhytomyr Polytechnic State University, Ukraine

3. doors 2024 organizers

The 4th edition of the doors was organized jointly by the Zhytomyr Polytechnic State University and the Academy of Cognitive and Natural Sciences (ACNS). ACNS, a non-governmental organization, is committed to developing researchers' knowledge of cognitive and natural sciences. Their mission involves advancing research, protecting individual rights and freedoms, and addressing professional, scientific, and social needs.

One of the noteworthy publications by ACNS is the *Journal of Edge Computing* (JEC, https://acnsci.org/jec), a peer-reviewed journal that delves into the realms of the Internet of Things, distributed systems, and edge computing. JEC focuses on scientific research on the utilization and implementation of edge computing across diverse domains such as education, science, medicine, and architecture.

ACNS also publishes such journals as *Educational Dimension* (https://acnsci.org/ed), *Educational Technology Quarterly* (https://acnsci.org/etq), *CTE Workshop Proceedings* (https://acnsci.org/cte). Notably, these journals cover a broad range of topics aligned with doors topics of interest:

- machine learning, deep learning and AI
- edge computing and edge devices
- distributed systems
- fault-tolerant computing
- UAV's
- IoT
- cloud and fog computing
- SMART house
- automated intelligent robotic platform
- biomedical systems
- GRID systems

4. Conclusion

The doors 2024 workshop was a resounding success, bringing together experts and professionals from various institutions and organizations to share their knowledge and ideas on edge computing. We express our gratitude to the Academy of Cognitive and Natural Sciences and Zhytomyr Polytechnic State University for their collaboration and support in the publishing of the *Journal of Edge Computing*.

We are immensely grateful to the authors and delegates who contributed to the success of the workshop by submitting their papers and participating actively in the discussions. We appreciate the efforts of the program committee members and the peer reviewers who provided their guidance, feedback, and support in improving the quality of the papers. Their valuable contributions and constructive critical comments helped to shape the content of the conference and made it a memorable experience for all participants.

We would like to acknowledge the developers and professional staff of the *Academy of Cognitive and Natural Sciences* (https://acnsci.org) and the *Not So Easy Science Education* platform (https://notso.easyscience.education) for providing us with the excellent and comprehensive conference management system that facilitated the smooth running of the workshop.

Since 2021, our workshop is **sponsored** by the CEUR Workshop Proceedings (CEUR-WS.org), the world best Diamond Open-Access proceedings publisher for Computer Science workshops. Long live CEUR-WS.org!

We believe that the presentations and discussions at the workshop have broadened our professional horizons and will serve as a catalyst for further research and innovation in the field of digital transformation in education. We look forward to meeting again in doors 2024 with renewed energy, enthusiasm, and a commitment to advancing the cause of edge computing.

Wireless technologies in IoT projects with distributed computing

Tetiana A. Vakaliuk[1,2,3,4], Oleksandr V. Andreiev[1], Oleksandr F. Dubyna[1], Oksana L. Korenivska[1] and Yevheniya O. Andreieva[1]

[1] *Zhytomyr Polytechnic State University, 103 Chudnivsyka Str., Zhytomyr, 10005, Ukraine*
[2] *Institute for Digitalisation of Education of the NAES of Ukraine, 9 M. Berlynskoho Str., Kyiv, 04060, Ukraine*
[3] *Kryvyi Rih State Pedagogical University, 54 Universytetskyi Ave., Kryvyi Rih, 50086, Ukraine*
[4] *Academy of Cognitive and Natural Sciences, 54 Universytetskyi Ave., Kryvyi Rih, 50086, Ukraine*

Abstract

When it comes to creating projects based on the use of the Internet of Things (IoT), wireless sensor networks are often used. The use of edge computing in IoT technology allows reducing system response delays to sensor output signals and increasing the network throughput. At the same time, a short-range sensor network can work locally without access to the Internet, while long-range networks, as a rule, require access to the Internet and use both edge and cloud computing. The article analyzes the possible options for wireless data transmission during the implementation of both short- and long-range IoT projects. Specific examples show the possibility of data transmission over a short distance using ESP-NOW technology, nRF24L01 radio modules and the creation of a local Wi-Fi access point. The range of sensor data transmission between microcontrollers is practically determined for each proposed option. The calculation of the range of the LoRa radio line is carried out for the real sensitivity values of the RFM95W receiver. The use of the Okamura-Hata radio wave propagation model is proposed in order to estimate the total signal loss in the LoRa radio line. The essence of edge computing with the combined use of digital and analog sensors is shown.

Keywords

IoT, wireless sensor network, monitoring system, edge devices

1. Introduction

Nowadays, a large number of projects are based on the use of the Internet of Things (IoT). For example, Sulistyawan et al. [1] describes the design of an IoT parking tracking system based on a NodeMCU ESP8266 microcontroller and an HC-SR04 ultrasonic sensor using a smartphone and a web application. Joshi and Patel [2] offer a smart parking system based on the ESP8266 Wi-Fi module and mobile Internet. In the wireless home automation system project, it is proposed to use the NodeMCU ESP8266 microcontroller to remotely control home appliances and the access system through a web browser and an Android application [3]. As another example, there is smart IoT-based home security system with an ESP32 microcontroller, that takes pictures of the room with a camera and transmits the information to the owner's smartphone when a motion sensor or smoke sensor is triggered. This system is described in [4].

The reliable operation of wireless sensor networks is directly related to the capacity of autonomous power sources of the sensor node. The most energy-consuming operation for sensor nodes is the transmission of data to the wireless environment. Therefore, energy-saving ways of transmission are a key factor in extending the sensors service life, as it is almost entirely dependent on the life of the power battery. Special communication protocols have been developed to solve this problem. To reduce energy consumption, the transmitters of the sensor nodes are usually being turned off, when

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine

tetianavakaliuk@gmail.com (T. A. Vakaliuk); oleks.andreyev@gmail.com (O. V. Andreiev); Dubyna1357@gmail.com (O. F. Dubyna); o.l.korenivska@gmail.com (O. L. Korenivska); Andreyeva.Yevheniya@gmail.com (Y. O. Andreieva)
http://acnsci.org/vakaliuk/ (T. A. Vakaliuk)
0000-0001-6825-4697 (T. A. Vakaliuk); 0000-0002-2601-1491 (O. V. Andreiev); 0000-0003-3448-6072 (O. F. Dubyna); 0000-0002-3735-7690 (O. L. Korenivska); 0009-0006-0822-1176 (Y. O. Andreieva)

no information transfer is required. Thus, only the simplest primary data processing that reduces the amount of transmitted information is performed on the sensor node. Therefore, preliminary processing of measurements is carried out during conducting edge computing. For example, when using analog sensors, the primary electrical signal is subject to analog-to-digital conversion, which can be performed directly in the microcontroller, if it has appropriate inputs for connecting analog signals.

To connect PAN (Personal Area Network) and LAN (Local Area Network) sensors to the global network, it is required to have a connection to a gateway, which can be implemented using technologies such as Ethernet, Wi-Fi and LoRaWAN. Recently, the use of global wireless networks, such as GSM, GPRS and LTE, has become widely used. These networks provide data transmission from sensors to remote cloud resources without the use of gateways. However, the literature research shows that insufficient attention has been paid to the analysis of data transfer options depending on the type of computation used to organize the IoT sensor network. Therefore, the purpose of this article is analyzing possible options for wireless data transmission in the implementation of IoT projects, both in edge and cloud computing.

2. Theoretical background

Both global and local networks are used to implement IoT projects. Wireless sensor networks (WSN) in which the distance between sensors does not exceed several dozens of meters, belong to wireless personal networks (WPAN).

We will consider the “machine to people” data transfer model in IoT, which is presented in figure 1.

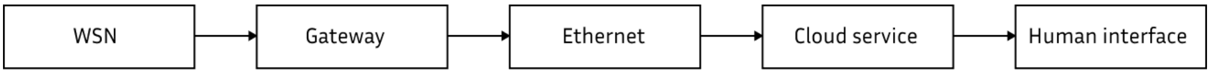


Figure 1: Model of data transmission in IoT.

The protocols that are used to transmit data between the nodes of WSN do not only differ from the protocol of the IoT global network, but also differ significantly from each other. For example, such protocols are ZigBee, Z-Wave and Bluetooth. These standards provide two-way communication between devices. Data is transmitted from the sensors to the network coordinator, which acts as a gateway and provides access to the external network. Through the coordinator, the sensor network also receives external control commands for the actuators that are part of the sensor network nodes. To perform collecting data from sensors and controlling actuators, sensor nodes contain microcontrollers, which must also perform communication functions. The coverage area of WSN can be significantly increased due to the fact that a number of communication protocols implies an opportunity of relaying messages from one sensor network node to another. In this regard, various WSN architectures with sufficiently large number of sensors and actuators with autonomous power are used when implementing IoT projects [5].

The use of cloud services for data storage and processing has lately gained significant development in IoT projects because of the integrated use of the wireless and M2M communications and the global Internet network. GSM, CDMA, LTE, WiMAX cellular networks provide access to the global Internet network with the possibility of using cloud computing. This provides direct communication between the "Internet of Things" and cloud services in wireless networks with a long range [6, 4].

A short-range wireless sensor network uses edge computing to display information within the reach of edge devices [7, 8]. For example, when implementing the project of a home weather station, it is not necessary to require access to the global network. The transmission of temperature and humidity measurement data can be carried out using communication protocols of wireless sensor networks over a short distance. The need for both increasing the throughput of the network and the ability to minimize the delay of transmission and data processing has led to the necessity of using edge computing in software based on IoT technology [9]. Quite a lot of attention is paid to the efficiency of data transmission using Wi-Fi technology in IoT projects. For example, in [7] “delays, loss of packets depending on their size, the ratio of service information to useful information in one transaction during data transmission using the MQTT and CoAP protocols are determined", and in [10] "the possibility of network overload situations and reducing its throughput is analyzed".

The maximum distance over which data packets can be transmitted between wireless sensor network nodes depends on many factors. To organize communication, first of all, it is necessary to choose the frequency range of the radio line. Nowadays, ZigBee, Z-Wave, Bluetooth, Wi-Fi and LoRaWAN standards, which use ISM frequency bands, are widely used to create IoT projects. The ISM band is available license-free in most countries, under the condition of limited transmitter's output power level. Changing the communication range is possible by changing both the sensitivity of the receiving device and by choosing the type and height of the antenna systems. At the same time, the communication range depends on the loss of the useful signal during propagation from the transmitter to the receiver. Bluetooth wireless technology is widely used, when it comes to creating home automation projects with the transmission of sensor data and control of devices at a short distance. This standard uses a frequency of 2.4 GHz and provides economical consumption of autonomous power sources [5].

It is convenient to use the nRF24L01 transceiver to create a wireless sensor network. It provides software selection of one of 125 ISM frequency channels in the 2.4-2.525 GHz range and has a printed antenna with a gain of 2 dBi. One receiver with a sensitivity of -82dBm and six transmitters can work simultaneously on the frequency of one channel. The transmitter power level is programmable from -18dBm to 0dBm in 6dBm increments. GFSK modulation of the frequency of the selected channel is used for data transmission. The SPI interface is used to connect the nRF24L01 to the microcontroller [11].

A gateway is used to get remote control of sensor network devices. The gateway is a central point of communication for all devices that work according to a certain protocol. It connects to the home Wi-Fi network. At the same time, home automation devices responsible for security, lighting, climate, etc. are connected to the gateway. As a result of the application of edge data computing from smart devices in the house, the Wi-Fi channel is not overloaded, and the response delay to the event is reduced compared to the use of cloud services. Sometimes Wi-Fi response delays can reach up to ten seconds [7].

Often there is a need to organize a gateway for access to the global Internet network several kilometers away from the sensor network. In this case, the LoRaWAN network protocol can be used for data transmission. This protocol has low energy consumption and uses LoRa broadband modulation at the physical level. The LoRa physical radio interface uses broadband radio signals with a big base B. This signal is highly resistant to interference. A CSS radio signal with bandwidth $BW = 125$, 250 or 500 kHz is used for data transmission. During digital data transmission, the radio signal base $B = BW \cdot T_{sym}$ is adaptively changed to ensure the required communication quality. This is achieved by changing the duration of the symbol $T_{sym} = 2^{SF}/BW$, which depends on the spreading factor of the radio signal (SF). This coefficient determines the data bits quantity transmitted during the time T_{sym} [12]. The LoRaWAN network protocol, in addition to the adaptive change in data transmission speed, also provides for changing the transmitter power for each edge device individually to ensure the specified quality of data transmission and economical use of autonomous power sources. At the same time, the radio range also changes.

The method of determining the maximum possible distance between WSN nodes involves the development of a distance calculation method or practical distance determination during the implementation of IoT projects. Therefore, when building a sensor network using a LoRa physical interface, it is important to have a method for calculating the range of transmission of data packets at a certain speed.

The maximum communication range R will be achieved under the condition that the power level of the received signal P_S is equal to the sensitivity of the receiver. The power level of the received signal in the radio line using radio waves of length λ, at the transmitter power level P_T, can be calculated by the formula [13, 14]:

$$P_S = P_T + G_T + G_R - L_{Loss}, \tag{1}$$

where L_{Loss} is signal loss during propagation from the transmitter to the receiver, G_T, G_R are coefficients of transmitting and receiving antennas.

Losses of the useful signal L_{Loss} are determined by specific conditions of radio wave propagation at a distance R. The multi-beam nature of radio waves propagation, the formation of shadow zones, multiple reflection and scattering of radio waves in the urban environment creates the phenomenon of intersymbol interference (ISI) in the transmission of digital data. Signal distortions caused by ISI can cause a deterioration in the quality of digital information transmission. Besides, there are signal losses during propagation in the atmosphere and due to the imperfection of the transceiver. All this leads to additional signal losses. The Okumura-Hata radio wave propagation model is well suited for estimating the total signal loss in the LoRa radio line, according to which the loss in the city is calculated by the expression [13]:

$$L_{50/Town} = 69.55 + 26.16 \lg f_{[MGz]} - 13.83 \lg h_B - a\left(h_M\right) + \left(44.9 - 6.55 \lg h_B\right) \cdot \lg R_{[km]}, \quad (2)$$

where $a\left(h_M\right)$ is correction factor.

For a small and medium-sized city, this coefficient is determined as follows:

$$a\left(h_M\right) = \left(1.11 \lg \left(f_{[MGz]}\right) - 0.7\right) h_M - \left(1.56 \lg \left(f_{[MGz]}\right) - 0.8\right). \quad (3)$$

To determine the signal power level at the receiver input, it is advisable to choose the maximum possible value of the radio signal attenuation.

3. Results

To analyze possible options for wireless data transmission, the authors of this article implemented examples of IoT projects and practically determined the range of sensor data transmission between microcontrollers for each proposed option.

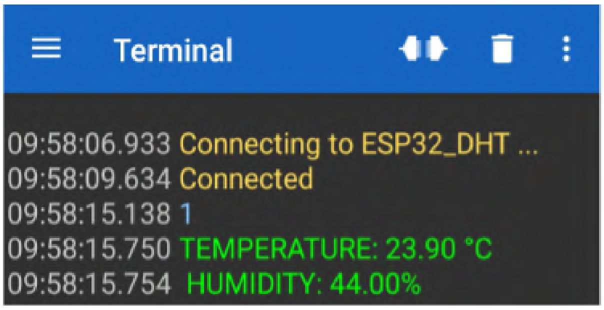
ESP8266 and ESP32 microcontrollers from Espressif Systems are widely used to create IoT projects. They both have a built-in Wi-Fi module. Moreover, the ESP32 microcontroller also supports the Bluetooth standard.

The results of the home weather station project using the ESP32 microcontroller and the DHT11 sensor are shown in figure 2. Data transmission of temperature and humidity measurements was carried out via the Bluetooth interface with display on the mobile device in the “Serial Bluetooth Terminal” application. The interface allows you to organize a home automation system to integrate or control electrical and electronic devices in the house at low cost. For instance, in the home weather station project, measurement data were displayed on the screen upon request sent to the microcontroller from the mobile application. The maximum range of communication between devices in the room was up to 30 meters.

The authors also implemented a project for transmitting temperature and humidity measurements using a DHT11 sensor between two nRF24L01 transceivers under the control of ESP8266 microcontrollers. The digital measurements data were analyzed both on the transmitting and receiving sides. As shown in figure 3, data transmission over the radio line is synchronous and error-free. It was practically determined that the range of communication is provided up to 30 m indoors and up to 100 m in the open area.

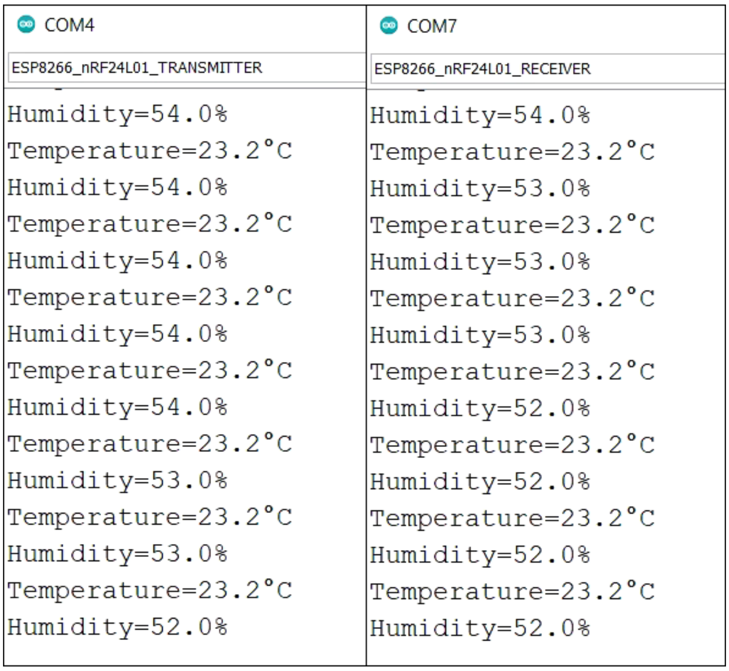
When using ESP8266 and ESP32 microcontrollers, it is also possible to create a local Wi-Fi access point with the display of sensor data on a webpage. The access to this webpage can be gained using any device, a laptop for example. The temperature and humidity measurement results can be displayed on a webpage using an IP address or DNS. This is shown in figure 4. Connection to the local Wi-Fi access point was carried out at a distance of at least 100 m. Using nRF24L01 transceivers allows to increase distance of measurement data transmission up to 200 m.

A home automation system of any standard can work completely locally without access to the Internet, which eliminates delays in the response of smart devices to events. For example, this is very important for motion sensors that should turn on the light after detecting a person.



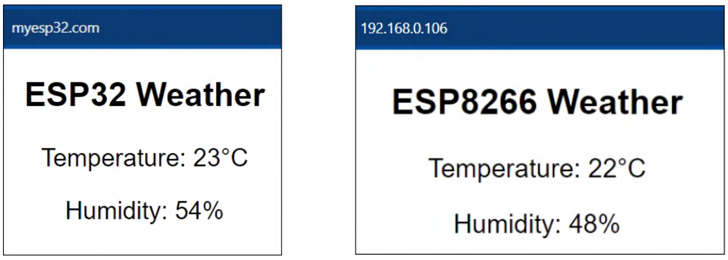
```
09:58:06.933 Connecting to ESP32_DHT ...
09:58:09.634 Connected
09:58:15.138 1
09:58:15.750 TEMPERATURE: 23.90 °C
09:58:15.754  HUMIDITY: 44.00%
```

Figure 2: Transmission of measurements via the Bluetooth interface.



```
COM4                              COM7
ESP8266_nRF24L01_TRANSMITTER      ESP8266_nRF24L01_RECEIVER
Humidity=54.0%                    Humidity=54.0%
Temperature=23.2°C                Temperature=23.2°C
Humidity=54.0%                    Humidity=53.0%
Temperature=23.2°C                Temperature=23.2°C
Humidity=54.0%                    Humidity=53.0%
Temperature=23.2°C                Temperature=23.2°C
Humidity=54.0%                    Humidity=53.0%
Temperature=23.2°C                Temperature=23.2°C
Humidity=54.0%                    Humidity=52.0%
Temperature=23.2°C                Temperature=23.2°C
Humidity=53.0%                    Humidity=52.0%
Temperature=23.2°C                Temperature=23.2°C
Humidity=53.0%                    Humidity=52.0%
Temperature=23.2°C                Temperature=23.2°C
Humidity=52.0%                    Humidity=52.0%
```

Figure 3: Transfer of temperature and humidity measurements between two transceivers nRF24L01.



myesp32.com

ESP32 Weather

Temperature: 23°C

Humidity: 54%

192.168.0.106

ESP8266 Weather

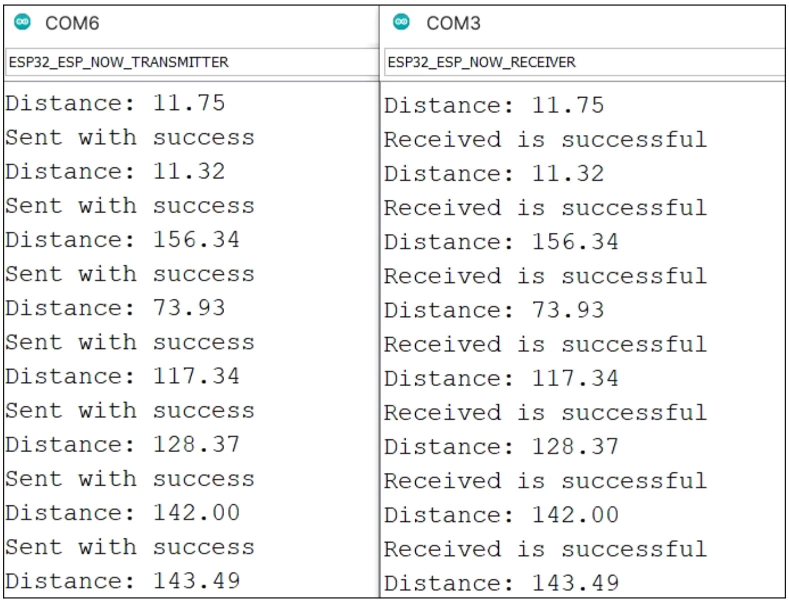
Temperature: 22°C

Humidity: 48%

Figure 4: Display of temperature and humidity measurements on a webpage.

ESP-NOW technology, developed by Espressif Systems, can also be used for two-way forwarding of data packets of up to 250 bytes with a transmission speed of no more than 1 Mbit/s between controllers. This technology is based on a simplified Wi-Fi protocol. At the same time, it is possible to organize a WSN in which communication between no more than 20 pairs of devices will be maintained, with the transmitter being informed about the success of forwarding packets. In order to send messages, you need to know the unique MAC address of the boards. If you need to collect data from several boards onto one, for example, to display data from several sensors on a web server, you can use the "one-slave – multiple-master" configuration. It is also possible to create a "one-master – multiple-slaves" configuration, when one board sends commands to different boards of microcontrollers of the ESP series [15].

The results of distance measurement transmission between ESP32 controllers using ESP-NOW technology are shown in figure 5. The distance was measured by an ultrasonic sensor HC-SR04. The distance in centimeters was calculated on the edge device. Received data were displayed on the laptop screen. During the practical implementation of the project, it was determined that the range of data transmission is up to 100 m.



```
Distance: 11.75
Sent with success
Distance: 11.32
Sent with success
Distance: 156.34
Sent with success
Distance: 73.93
Sent with success
Distance: 117.34
Sent with success
Distance: 128.37
Sent with success
Distance: 142.00
Sent with success
Distance: 143.49
```

```
Distance: 11.75
Received is successful
Distance: 11.32
Received is successful
Distance: 156.34
Received is successful
Distance: 73.93
Received is successful
Distance: 117.34
Received is successful
Distance: 128.37
Received is successful
Distance: 142.00
Received is successful
Distance: 143.49
```

Figure 5: Ultrasonic distance meter using technology ESP-NOW.

Transmission of sensor data to a server, cloud service or edge user can be done through a gateway based on ESP32 or ESP8266 microcontrollers. For instance, the authors of the article created a gateway with a connection to the Wi-Fi access point based on the ESP8266 microcontroller. Measurements data from indoor air quality control sensors were transferred to this gateway using nRF24L01 transceivers. Afterwards, those measurement results were displayed at the ThingSpeak cloud service. A DHT11 digital sensor was used to measure temperature and humidity. An MQ-2 analog sensor was used to determine the concentration of hydrocarbon gases. The edge computing of measurement data was carried out in the ESP8266 microcontroller of the transmission part of the radio line. In order to read digital data from the DHT11 sensor, a library supporting the SDA interface was used. The gas concentration was determined by the voltage at the output of the MQ-2 sensor. The built-in 10-bit ADC of the ESP8266 board was used to get digital measurement data of output voltage. The relative internal resistance of the sensor was calculated based on the measured voltage.

This resistance value was used to estimate the gas concentration value based on the calibration characteristic of the sensor. The gas concentration value was sent to the nRF24L01 transceiver through the SPI interface. At the receiving end of the radio line, from the output of another nRF24L01transciever, the gas concentration value was input to the ESP8266 microcontroller via the SPI interface. On the same controller, a gateway was created for transmitting gas concentration measurements to the ThingSpeak by connecting to a Wi-Fi access point. The change in measurements of air quality control sensors at a certain time interval, which was output to the ThingSpeak cloud service, is shown in figure 6.

It should be noted, that the MQ-2 sensor does not determine the type of gas. It only reacts to an increase in the concentration of liquefied gas and other carbohydrates in the air. To simulate an increase in LPG content in the air, a gas lighter refill aerosol can was used. Figure 6 shows that the sensor responds very well to an increase in the concentration of this gas. Blowing on the sensor also leads to an increase in CO concentration. In addition, a simultaneous increase in temperature and air humidity according to the measurements of the DHT11 sensor was also noted.

Therefore, the graphical display of changes in controlled parameters in the IoT cloud service ThingS-

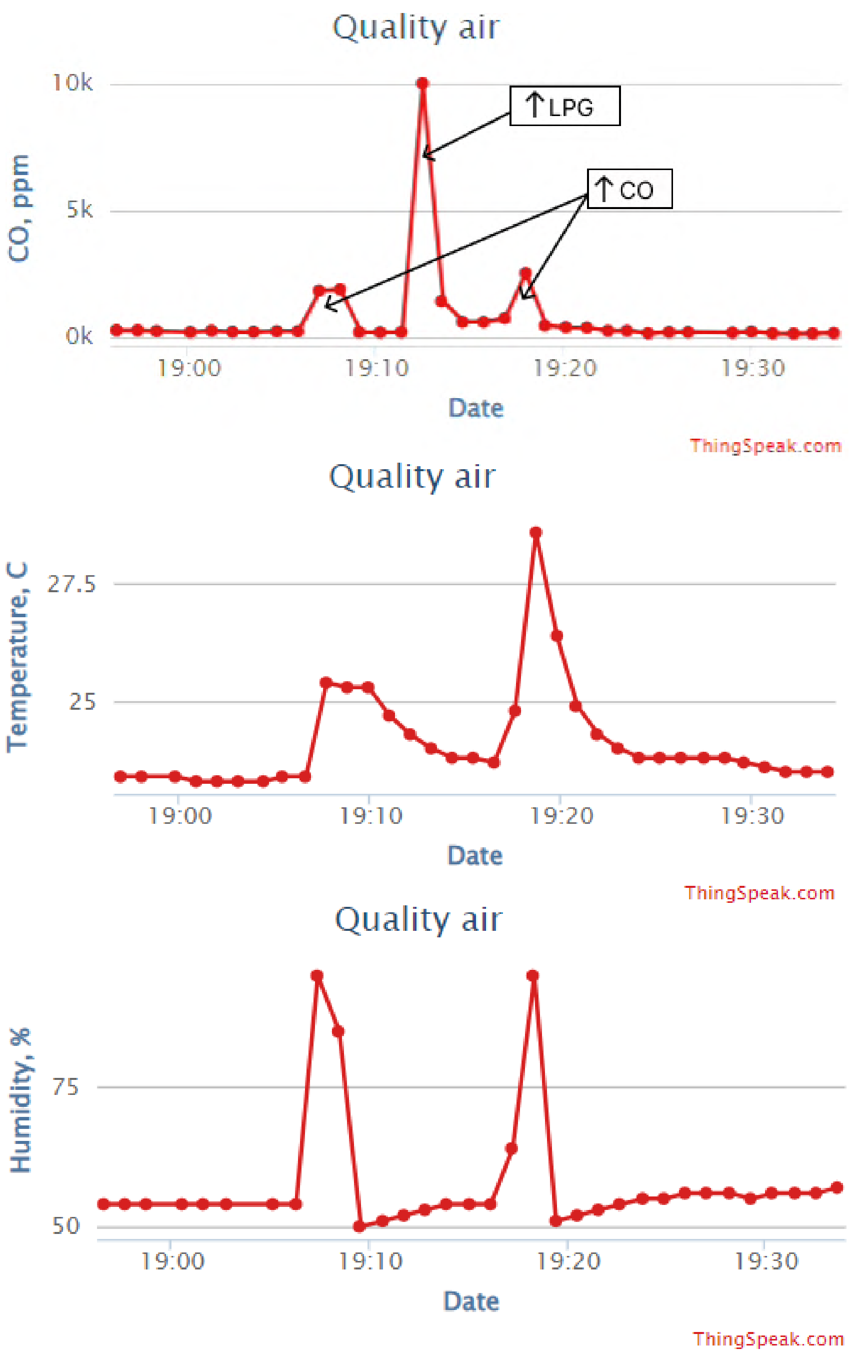


Figure 6: Display of measurements of air quality control sensors in the cloud service ThingSpeak.

peak allows you to carry out both daily monitoring of changes in parameters and statistical analysis of measurements. However, a necessary condition for the correct operation of this service is the implementation of a delay of at least 20 seconds between the transmission of measurements of each channel [16]. An attempt to reduce the delay time or to eliminate it at all, led to a disruption of the service. In order to display the dynamic change of the controlled parameter, the authors developed an IOT project for measuring CO concentration in the room using the Blynk cloud service. The display of CO concentration in Blynk.Console is shown in figure 7.

In order to determine the coverage area of a wireless sensor network using the LoRaWAN communication protocol, the range of data transmission over the LoRa radio line at a specified speed was calculated. The range of data transmission over the LoRa radio link is determined by the selected uplink (UL) and downlink (DL) parameters. These parameters in Europe are shown in table 1.

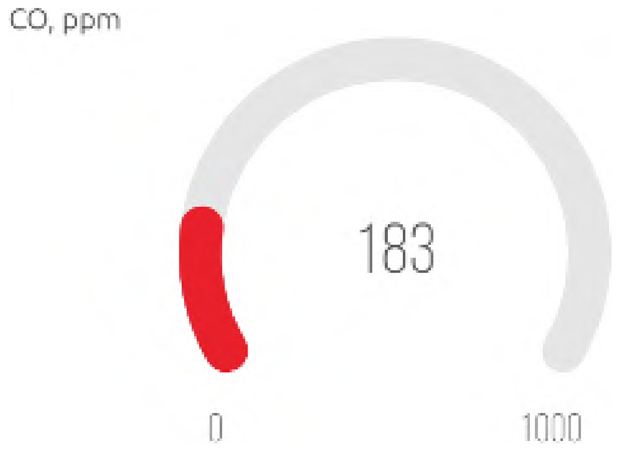


Figure 7: Display of measurements of the MQ-2 sensor in the “Blynk” cloud service.

Table 1
Parameters of the LoRa standard in Europe.

Parameter	Band1, MHz	BW UL, kHz	BW DL, kHz	P_T UL, dBm	P_T DL, dBm	SF	Modulation
Value	863 – 870	125/250	125	2-14	14	7-12	LORA, GFSK, MSK

Typical values of LoRa modem parameters for the frequency of 868 MHz are given in table 2[17].

Table 2
Example LoRaTM modem performances, 868 MHz.

Bandwidth (kHz)	Spreading Factor	Coding rate	Nominal Rb (bps)	Sensitivity indication (dBm)
125	6	4/5	9380	-118
125	12	4/5	293	-136

The approximate communication range R and the data transmission rate R_b, which is provided by a LoRa radio line operating at a frequency of 868 MHz and $BW = 125$ kHz were calculated. The omnidirectional antenna and the transmitters with a power level of 14 dBm were used for conducting the calculations. The communication range according to expressions (1)-(3), under the condition that $h_B = 30$ m and $h_M = 2$ m, was calculated. The speed of information transmission without an interference-resistant code $(CR = 1)$ is defined by the expression $R_b = SF \cdot BW/2^{SF}$. The use of an interference-resistant code reduces the speed of information transmission according to the value of the CR parameter. The results of the calculation of the communication range and data transmission speed on the LoRa radio line for the value of $CR = 4/5$ are shown in table 3.

Table 3
The results of calculating the range and speed of data transmission over the LoRa radio line.

SF	6	7	8	9	10	11	12
R,km	3	4,5	5,5	6,7	8,1	8,7	10
R_b, bps	9375	5469	3125	1758	977	537	293

As can be seen from the table 3, increasing the value of SF leads to increase in the communication range with a simultaneous decrease in the information rate of data transmission.

4. Conclusions

The use of edge computing in IoT technology allows reducing system response delays to sensor output signals and increasing the throughput of the network. This type of distributed computing is always carried out in close proximity to the edge devices. Therefore, edge computing takes place in the creation of sensor networks of both short- and long-range. At the same time, a short-range sensor network can work locally without access to the Internet, while long-range networks, as a rule, require access to the Internet and use both edge and cloud computing.

The article analyzes the possible options for wireless data transmission during the implementation of both short- and long-range IoT projects. Using real examples, the authors show the possibility of data transmission over a short distance using ESP-NOW technology, nRF24L01 radio modules and the creation of a local Wi-Fi access point. The results of the home weather station project using Bluetooth technology were presented. The range of sensor data transmission between microcontrollers was practically determined for each proposed option.

The calculation of the range of the LoRa radio line was carried out for the real sensitivity values of the RFM95W receiver. To estimate the total signal loss in the LoRa radio line, the use of the Okamura-Hata radio wave propagation model was proposed. The obtained values of the radio line range make it possible to more accurately determine the WSN coverage area when using the LoRaWAN communication protocol.

The essence of edge computing with the combined use of digital and analog sensors was shown. An example of data transmission of the air quality control system through a gateway based on an ESP8266 microcontroller with a graphical display of measurements in the IoT cloud service ThingSpeak and Blynk was provided.

In further research, it is planned to practically determine the maximum communication range between WSN nodes by implementing IoT projects using RFM95W modems.

5. Author contributions

Conceptualization, formulation of tasks – Tetiana Vakaliuk; air quality control projects, analysis of results – Oksana Korenivska, Oleksandr Dubyna; distance meter project using ESP-NOW technology, analysis of results – Oleksandr Andreiev; conceptual analysis - Tetiana A. Vakaliuk, Oleksandr Andreiev, method of calculating the radio line range, analysis of results – Yevheniya Andreieva; writing – original draft preparation and editing – Tetiana Vakaliuk, Yevheniya Andreieva.

All the authors have read and agreed to the published version of this manuscript.

References

[1] V. N. Sulistyawan, N. A. Salim, F. G. Abas, N. Aulia, Parking Tracking System Using Ultrasonic Sensor HC-SR04 and NODEMCU ESP8266 Based IoT, IOP Conference Series: Earth and Environmental Science 1203 (2023) 012028. doi:10.1088/1755-1315/1203/1/012028.

[2] A. Joshi, J. Patel, Smart parking system using esp8266 wi-fi module, International Journal of Advanced Research in Science Communication and Technology 3 (2023) 88–95. doi:10.48175/IJARSCT-12015.

[3] G. Rajeshkumar, P. Rajesh Kanna, S. Sriram, S. Sadesh, R. Karunamoorthi, P. Mahudapathi, Home Automation System Using Nodemcu (ESP8266), in: R. Buyya, S. Misra, Y.-W. Leung, A. Mondal (Eds.), Proceedings of International Conference on Advanced Communications and Machine Intelligence, Springer Nature, Singapore, 2023, pp. 293–302. doi:10.1007/978-981-99-2768-5_28.

[4] O. L. Korenivska, T. M. Nikitchuk, T. A. Vakaliuk, V. B. Benedytskyi, O. V. Andreiev, IoT monitoring system for microclimate parameters in educational institutions using edge devices, in: T. A. Vakaliuk, S. O. Semerikov (Eds.), Proceedings of the 3rd Edge Computing Workshop, Zhytomyr,

Ukraine, April 7, 2023, volume 3374 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2023, pp. 66–80. URL: https://ceur-ws.org/Vol-3374/paper05.pdf.

[5] A. McEwen, H. Cassimally, Designing the Internet of Things, Wiley, 2014. URL: https://madsg.com/wp-content/uploads/2015/12/Designing_the_Internet_of_Things.pdf.

[6] ETSI TS 102 690 V2.1.1 (2013-10) "Machine-to-Machine communications (M2M); Functional architecture", 2013. URL: https://www.etsi.org/deliver/etsi_ts/102600_102699/102690/02.01.01_60/ts_102690v020101p.pdf.

[7] E. Malokhvii, H. Molchanov, Research of data transmission protocols in the conditions of the Internet of Things, Control, Navigation and Communication Systems. Academic Journal 1 (2022) 66–74. doi:10.26906/SUNZ.2022.1.066.

[8] O. L. Yershova, T. V. Tomashevska, Peripheral Computations: The Basis for Data Processing in Internet of Things, Scientific Bulletin of the National Academy of Statistics, Accounting and Audit (2021) 97–103. doi:10.31767/nasoa.4-2020.11.

[9] O. V. Talaver, T. A. Vakaliuk, Reliable distributed systems: review of modern approaches, Journal of Edge Computing 2 (2023) 84–101. doi:10.55056/jec.586.

[10] N. I. Pravorska, N. V. Hripinska, Development Of Sensor For Internet Of Things Data Transferring, Bulletin of Khmelnytsky National University. Technical sciences 6 (2018) 193–197. doi:10.31891/2307-5732-2018-267-6(2)-193-197.

[11] Single chip 2.4 GHz Transceiver nRF24L01, 2006. URL: https://www.sparkfun.com/datasheets/Components/nRF24L01_prelim_prod_spec_1_2.pdf.

[12] LoRa and LoRaWAN: A Technical Overview, Semtech, 2019. URL: https://lora-developers.semtech.com/uploads/documents/files/LoRa_and_LoRaWAN-A_Tech_Overview-Downloadable.pdf.

[13] A. Pinto-Erazo, L. Suárez-Zambrano, F. Cuzme-Rodríguez, E. Jaramillo-Vinueza, Study Case: LPWAN Propagation Power Estimation in Outdoor Scenarios, Based on the Okumura-Hata Model, in: V. Robles-Bykbaev, J. Mula, G. Reynoso-Meza (Eds.), Intelligent Technologies: Design and Applications for Society, Springer Nature Switzerland, Cham, 2023, pp. 113–123. doi:10.1007/978-3-031-24327-1_10.

[14] T. A. Vakaliuk, O. V. Andreiev, O. F. Dubyna, T. M. Nikitchuk, I. V. Puleko, Detection of the signals of the terrestrial radar stations by spacecraft with a passive synthesis of the antenna aperture, Radio Electronics, Computer Science, Control (2023) 13. doi:10.15588/1607-3274-2023-2-2.

[15] ESP-NOW wireless communication protocol, 2023. URL: https://www.espressif.com/en/solutions/low-power-solutions/esp-now.

[16] ThingSpeak for IoT Projects, 2023. URL: https://thingspeak.com.

[17] SX1276/77/78/79 -137MGz to 1020 MGz Low Power Long Range Transceiver, 2015. URL: https://html.alldatasheet.com/html-pdf/800239/SEMTECH/SX1276/891/3/SX1276.html.

Edge computing applications: using a linear MEMS microphone array for UAV position detection through sound source localization

Andrii V. Riabko[1], Tetiana A. Vakaliuk[2,3,4,5], Oksana V. Zaika[1], Roman P. Kukharchuk[1] and Valerii V. Kontsedailo[6]

[1] *Oleksandr Dovzhenko Hlukhiv National Pedagogical University, 24 Kyivska Str., Hlukhiv, 41400, Ukraine*
[2] *Zhytomyr Polytechnic State University, 103 Chudnivsyka Str., Zhytomyr, 10005, Ukraine*
[3] *Institute for Digitalisation of Education of the NAES of Ukraine, 9 M. Berlynskoho Str., Kyiv, 04060, Ukraine*
[4] *Kryvyi Rih State Pedagogical University, 54 Universytetskyi Ave., Kryvyi Rih, 50086, Ukraine*
[5] *Academy of Cognitive and Natural Sciences, 54 Universytetskyi Ave., Kryvyi Rih, 50086, Ukraine*
[6] *Inner Circle, Nieuwendijk 40, 1012 MB Amsterdam, Netherlands*

Abstract

This study explores the use of a microphone array to determine the position of an unmanned aerial vehicle (UAV) based solely on the sound of its engines. The accuracy of localization depends crucially on the arrangement of the microphones. The study also considers a mathematical model of pulse density modulation for a digital MEMS microphone. It demonstrates the frequency dependence of the efficiency of a differential array of first-order microphones. Based on this frequency dependence of directivity and the instability model of the microphone parameters, a rational operating frequency range for the normal functioning of the microphone array can be established. The study proposes a model of a linear microphone array based on MEMS omnidirectional microphones. With a specific geometrical arrangement, this array produces a bidirectional pattern, which can be easily transformed into a unidirectional pattern using specialized algorithms or hardware (e.g., ADAU1761 codecs).

Keywords

edge computing, UAV, sound source localization, MEMS microphone, microphone array, frequency, directivity

1. Introduction

Determining the position of a UAV (Unmanned Aerial Vehicle) by the sound of its engines can be important for several reasons. In military or security applications, being able to identify and locate UAVs by their engine sounds can help in detecting potential threats, including hostile drones or unauthorized surveillance. Sound-based localization can aid in the development of countermeasures to mitigate the risks posed by UAVs in sensitive areas.

Sound-based UAV detection can complement existing air traffic management systems, providing additional situational awareness for managing airspace and preventing collisions with manned aircraft. In search and rescue operations or in case of lost or malfunctioning drones, sound-based tracking can assist in locating and recovering UAVs.

In conservation efforts, it can help monitor UAVs used for illegal activities like poaching or wildlife disturbance. In urban areas or regions with dense UAV traffic, sound-based tracking can be useful for enforcing regulations related to UAV flight paths, altitudes, and no-fly zones. For protecting privacy,

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine

ryabko@meta.ua (A. V. Riabko); tetianavakaliuk@acnsci.org (T. A. Vakaliuk); ksuwazaika@gmail.com (O. V. Zaika); kyxap4yk1@ukr.net (R. P. Kukharchuk); valerakontsedailo@gmail.com (V. V. Kontsedailo)
http://irbis-nbuv.gov.ua/ASUA/0051396 (A. V. Riabko); https://acnsci.org/vakaliuk/ (T. A. Vakaliuk); http://pfm.gnpu.edu.ua/index.php/struktura1/2015-04-01-14-50-26 (O. V. Zaika); http://irbis-nbuv.gov.ua/ASUA/0076404 (R. P. Kukharchuk)
0000-0001-7728-6498 (A. V. Riabko); 0000-0001-6825-4697 (T. A. Vakaliuk); 0000-0002-8479-9408 (O. V. Zaika); 0000-0002-7588-7406 (R. P. Kukharchuk); 0000-0002-6463-370X (V. V. Kontsedailo)

sound-based detection can help identify UAVs flying near private properties, providing a means to take legal action against intrusive drones.

Studying the acoustic signatures of UAVs can aid in research and development efforts to design quieter and more environmentally friendly drones. During natural disasters or emergencies, knowing the positions of UAVs, such as those used for aerial surveys or damage assessment, can assist in coordinating response efforts. Sound-based UAV detection can be employed in border control to monitor and respond to unauthorized drone incursions.

As drone delivery and urban air mobility concepts develop, sound-based localization can contribute to managing UAV traffic in urban environments. While sound-based UAV localization offers several advantages, it also has limitations, such as accuracy challenges in noisy environments and the need for specialized equipment. Therefore, it is often used in conjunction with other tracking and detection methods, such as radar, visual recognition, and GPS, to provide comprehensive situational awareness and enhance safety and security in various applications.

The goal of our work is to develop a software and hardware system for capturing hardware-synchronized sound using digital MEMS microphones (Microelectromechanical Systems, MEMS) for further use in sound source localization systems. This system is intended for further use in sound source localization systems, marking a significant advancement in the field of edge computing.

2. Theoretical background

Over the past few decades, acoustic source localization has emerged as a focal point of interest within the research community [1, 2]. Most studies of sound source identification are based on the analysis of the physiological mechanism of human hearing [3, 4]. It is common practice to use arrays of microphones [5]. An actual problem is acoustic beam formation for sound source localization and its application [6].

Microphone array processing represents a well-established methodology employed in the estimation of sound source direction. In a groundbreaking contribution by Yamada et al. [7], they introduce an innovative approach referred to as Multiple Triangulation and Gaussian Sum Filter Tracking (MT-GSFT). This advanced technique adeptly derives the precise location of sound sources through triangulation, utilizing microphone arrays seamlessly integrated into a fleet of multiple drones [7]. The domain of speech signal processing encompasses several critical areas, and among them, multiple sound source localization (SSL) stands out as a notable and relevant field. A notable contribution to this field comes from Firoozabadi et al. [8], who introduced a two-step approach for the localization of multiple sound sources in three dimensions (3D). This method relies on the precise estimation of time delays (TDE) and strategically leverages distributed microphone arrays (DMA) to enhance the accuracy and effectiveness of the localization process [8].

Sasaki et al. [9] present a method designed to map the 3D coordinates of a sound source by leveraging data gathered from an array of microphones, with each microphone providing an autonomous directional estimate. Additionally, LiDAR technology is employed to create a comprehensive 3D representation of the surroundings and accurately determine the sensor's position with six degrees of freedom (6-DoF).

Catalbas et al. [10] conduct a comparative analysis, assessing the effectiveness of generalized cross-correlation techniques in contrast to noise reduction filters concerning the estimation of sound source trajectory. Throughout the entire movement, they calculate the azimuth angle between the sound source and the receiver. This calculation relies on the parameter of Interaural Time Difference (ITD) to determine the azimuth angle. They then evaluate the accuracy of the estimated delay using various types of Generalized Cross-Correlation (GCC) algorithms for comparison.

It is possible for unmanned aerial vehicles (UAVs) to use audio information to compensate for poor visual information. Hoshiba et al. [11] developed a microphone array system built into the UAV to localize the sound source in flight. They developed the Spherical Microphone Array System (SMAS), consisting of a microphone array, a stable wireless network communication system, and intuitive visualization tools.

Tachikawa et al. [12] introduced an innovative approach that involves estimating positions by utilizing a modified variant of the convex clustering method in conjunction with sparse coefficients estimation. Additionally, they put forth a technique for constructing a well-suited monopole dictionary, which is based on coherence, ensuring that the convex clustering-based method can accurately estimate the distances of sound sources. The study involved conducting a series of numerical and measurement experiments aimed at assessing the effectiveness and performance of this novel methodology.

When dealing with multiple sound sources, establishing a reliable data association between localization information and the corresponding sound sources becomes paramount for achieving optimal performance. To address the challenges posed by data association uncertainty, Wakabayashi et al. [13] extended the Global Nearest Neighbor (GNN) approach, introducing a modified version known as GNN-c, specifically tailored to meet the real-time and low-latency requirements of drone audio applications. The outcome of their efforts showcases a system capable of accurately estimating the positions of multiple sound sources, achieving an impressive accuracy level of approximately 3 meters.

Many acoustic image-based sound source diagnosis systems suffer from spatial stationary limitations, making it challenging to integrate information from various capture positions, thereby leading to unreliable and incomplete diagnostics. In their paper, Carneiro and Berry [14] introduce a novel measurement methodology called Acoustic Imaging Structure From Motion (AISFM). This approach utilizes a mobile spherical microphone array to create acoustic images through beamforming, seamlessly integrating data from multiple capture positions. Their method is not only proposed but also meticulously developed and rigorously validated, offering a promising solution to enhance the accuracy and comprehensiveness of sound source diagnostics.

In a research conducted by Kita and Kajikawa [15] a sound source localization (SSL) technique is introduced, specifically designed for the localization of sources situated within structures, including mechanical equipment and buildings.

The registration of acoustic signals with cross-shaped antennas is widely discussed in the literature [16].

Advanced signal processing methods involving multiple microphones can enhance noise resilience. However, as the quantity of microphones employed escalates, the computational overhead rises concomitantly. This, in turn, curtails response time and hinders their extensive adoption across various categories of mobile robotic platforms [17]. Within the realm of robot audition, sound source localization (SSL) holds a pivotal role, serving as a fundamental component. SSL empowers a robotic platform to pinpoint the origin of sound using auditory cues exclusively. Its significance extends beyond mere sound localization, as it significantly influences other facets of robot audition, including source separation. Moreover, SSL contributes to elevating the quality of human-robot interaction by augmenting the robot's perceptual prowess [18].

In general, machine learning is widely used in acoustics [19, 20]. In the realm of human-robot interaction, He et al. [21] have introduced a pioneering approach. Their proposal involves harnessing neural networks for the simultaneous detection and localization of multiple sound sources. This innovative method represents a departure from conventional signal processing techniques by offering a distinct advantage: it necessitates fewer stringent assumptions about the environmental conditions, thereby enhancing its adaptability and effectiveness [21]. Ebrahimkhanlou and Salamone [22] have put forth an advanced methodology for localizing acoustic emissions (AE) sources within metallic plates, especially those with intricate geometric features like rivet-connected stiffeners. This innovative approach leverages two deep learning techniques: a stack of autoencoders and a convolutional neural network (CNN), strategically employed to enhance the accuracy and precision of the localization process [22].

In their pioneering work, Adavanne et al. [23] have introduced an innovative solution – a convolutional recurrent neural network (CRNN) – designed to address the intricate task of joint sound event localization and detection (SELD) within three-dimensional (3-D) space. This method represents a significant advancement in the field, enabling the simultaneous identification and spatial localization of multiple overlapping sound events with remarkable precision.

Let's summarize the theoretical review. Localizing a sound source means determining the direction or location from which a sound is emanating. There are several algorithms and techniques used for sound source localization, and the choice of method often depends on the specific application and available hardware. Here are some commonly used algorithms. Time Difference of Arrival (TDOA) is based on measuring the time it takes for a sound to reach multiple microphones. By comparing the time differences, it's possible to triangulate the source's position. Cross-correlation or beamforming techniques are often used to calculate the time differences accurately. Generalized Cross-Correlation (GCC) is a technique used in conjunction with TDOA. It involves cross-correlating the signals from two or more microphones to find the delay between them. GCC-PHAT (GCC with Phase Transform) is a commonly used variant that works well in reverberant environments. Steering Vector Methods are commonly used in microphone arrays or beamforming applications [24]. They estimate the direction of arrival (DOA) by analyzing the phase differences between signals received by different microphones. Popular algorithms include Multiple Signal Classification (MUSIC) and Estimation of Signal Parameters via Rotational Invariance Techniques (ESPRIT). Acoustic Intensity Methods measure the sound intensity at multiple microphone positions and use this information to estimate the source direction. The Steered Response Power (SRP) algorithm is an example of this approach. Machine learning and deep learning techniques, such as neural networks and support vector machines, can be used to train models for sound source localization. These models can take input from multiple microphones and learn to predict the source location based on training data. Particle filtering is a probabilistic method that estimates the source location using a Bayesian filtering approach. It is useful when dealing with complex and dynamic environments. Some methods use time-frequency analysis techniques like the Short-Time Fourier Transform (STFT) or Wavelet Transform to analyze the spectral content of audio signals and infer the source location. In mobile sound source localization, the Doppler effect can be used to estimate the source's speed and direction based on the frequency shift in the received signal. Many practical systems use a combination of the above techniques to improve accuracy and robustness, especially in real-world scenarios with noise and reverberation.

The choice of algorithm depends on factors like the number and arrangement of microphones, environmental conditions, computational resources, and the desired level of accuracy. Different applications, such as robotics, audio conferencing, surveillance, and hearing aids, may employ different algorithms tailored to their specific requirements.

Determining the position of a UAV (Unmanned Aerial Vehicle) based solely on the sound of its engines can be challenging but is feasible using a combination of sound source localization techniques and signal processing. Here's a high-level overview of the process:

1. Microphone Array Setup: Set up a microphone array on the ground. The microphones should be strategically placed to capture the UAV's sound from different angles. The arrangement of microphones plays a crucial role in accurate localization. The response of microphone arrays depends, first of all, on the number of microphones working on the array [25].
2. Sound Data Collection: Record the sound generated by the UAV's engines as it flies overhead. Ensure that the recording system has a high sampling rate to capture the sound accurately.
3. Time Delay of Arrival (TDOA): Analyze the recorded audio data to calculate the time delay of arrival (TDOA) of the sound at each microphone. TDOA is the time difference between when the sound reaches different microphones. This information is critical for triangulation.
4. Triangulation: Use the TDOA data from multiple microphones to triangulate the UAV's position. Several algorithms, such as multilateration or beamforming, can help estimate the UAV's coordinates based on the TDOA information.
5. UAV Sound Signature: To improve accuracy, consider using machine learning techniques to create a database of UAV sound signatures. This involves training a model to recognize the unique sound characteristics of different UAVs. When a new sound recording is obtained, the model can help identify the specific UAV type.
6. Integration with Other Sensors: For real-time tracking, integrate sound-based localization with other sensors like GPS, radar, or visual cameras. This fusion of data sources can provide more accurate and robust positioning.

7. Calibration and Testing: Regularly calibrate and test the microphone array and signal processing algorithms to ensure accurate and reliable results.

It's important to note that the accuracy of sound-based UAV localization depends on various factors, including the UAV's altitude, speed, engine type, and background noise. Additionally, environmental conditions, such as wind and temperature, can affect sound propagation and localization accuracy. Therefore, this method may work best in controlled environments or in conjunction with other tracking methods for enhanced precision and reliability.

3. Research methods

The goal of our work is to develop a software and hardware system for capturing hardware-synchronized sound using digital MEMS microphones (Microelectromechanical Systems, MEMS) for further use in sound source localization systems.

Despite the fact that the use of radar equipment has become part of everyday practice when monitoring UAVs, there is some interest in assessing the possibility of using airborne acoustic signals for this purpose. The above applies mainly to receiving hydroacoustic antennas, i.e. to conditions when the speed of the source is much lower than the speed of sound $M = v/c \ll 1$. In the case of receiving air-acoustic signals propagating at a speed of sound significantly lower than that of hydroacoustic signals in water and created by fairly fast moving sources (passenger cars on autobahns, racing cars, the movement of airliners along runways during takeoff and landing, UAVs), there is a different situation. Research into the features of recording these signals with phased arrays remains relevant, since on their basis data can be obtained on the current coordinates and speed of movement of a moving object. The purpose of this work is to analyze the angular dependencies in the signal at the output of a receiving air-acoustic antenna and those qualitative changes in their nature that are introduced due to a combination of factors such as the Doppler effect and the sharp directivity of the antenna array.

A special case is considered, which is widespread in everyday practice, when the trajectory of an object is rectilinear, lies in a horizontal plane, close and parallel to the Earth's surface, and the speed of its movement is constant.

As previously stated, the arrangement of microphones plays a crucial role in accurate localization. The purpose of the study is to find the optimal configuration of a microphone array for localizing a moving sound source (UAV).

Directivity is the sensitivity of a microphone to sound depending on the direction or angle from which the sound is coming. Directionality or sound pickup angle is considered to be the area of possible location of the sound signal source, within which there is no significant loss of microphone efficiency. Microphones use different directivity characteristics. They are most often depicted as polar diagrams. This is done to graphically display sensitivity variations around the microphone over a 360-degree range, where the microphone is the center of the circle and the angular reference point is placed in front of the microphone. The polar pattern shows how a microphone's sensitivity to a sound signal depends on the location of its source.

Microphone arrays are an array of several microphones combined by joint digital signal processing. Microphone arrays provide the following advantages over single-channel systems: 1) directionality of sound reception; 2) noise suppression of point sources; 3) suppression of non-stationary environmental noise; 4) partial weakening of reverberation; 5) the possibility of spatial localization of the sound source; 6) the ability to accompany a moving point sound source.

A microphone array is one of the types of directional microphones, implemented as a set of sound receivers operating in concert (in phase or with certain phase delays). Geometrically, gratings can be implemented in different configurations – one-dimensional (linear, arc-shaped), two-dimensional (flat, spherical), three-dimensional, spiral, with uniform or non-equidistant pitch. The array's radiation pattern is created by changing the ratio of phase delays for different channels (in the simplest case, an in-phase array with a fixed position of the main lobe; in more complex and expensive implementations, a scanning system). The implementation of phase delays can be hardware (for example, on analog delay lines) or software (digital).

The basic microphone array structures are Broadside and Endfire (figure 1).

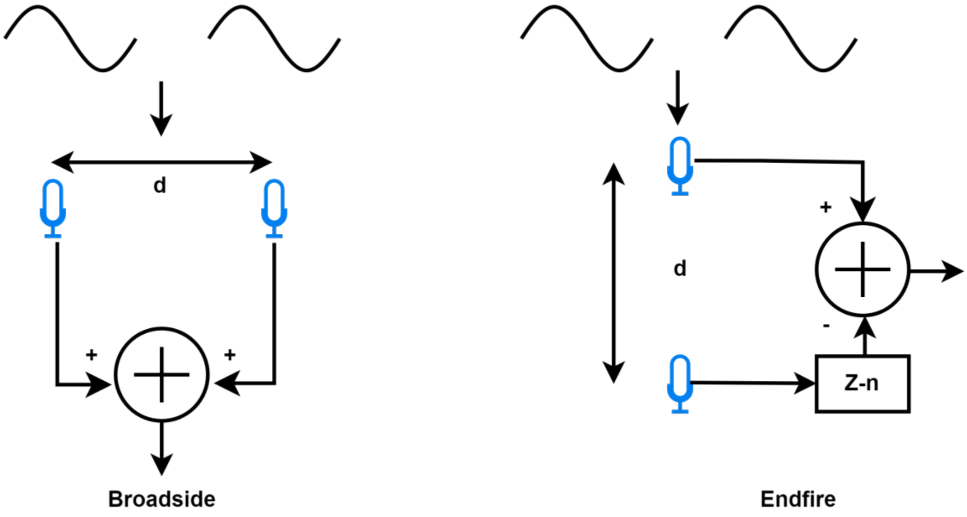


Figure 1: Basic array structures.

These structures use omnidirectional microphones (microphones that, regardless of their orientation, receive signals from any direction). The figure 2 shows signal reception versus direction for various frequencies with a single omnidirectional microphone. For one microphone, frequency invariance is observed.

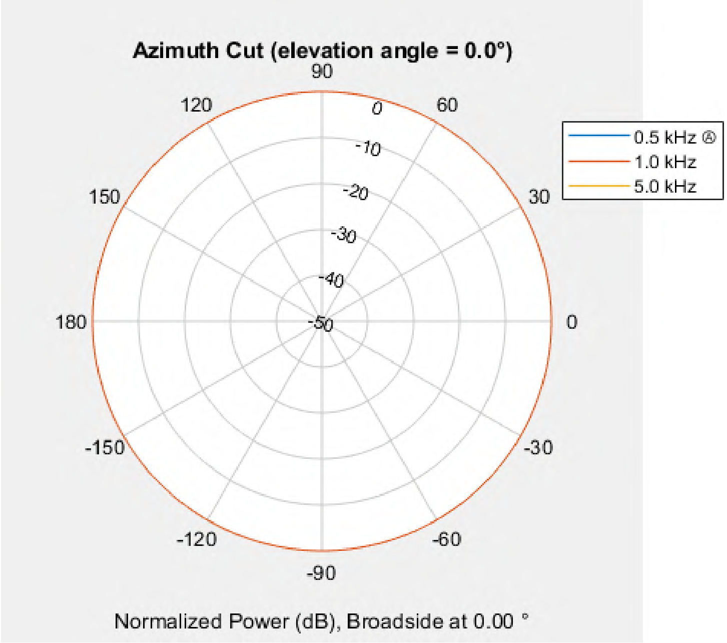


Figure 2: Dependence of signal reception on direction by one omnidirectional microphone for frequencies 500 Hz, 1 and 5 kHz.

The Broadside structure is an array of omnidirectional microphones positioned perpendicular to the direction of the desired signal. Such arrays have an axis of symmetry, relative to which the sound is released without attenuation both “in front” of the array and “behind”. Such structures are widely used in applications where sound pressure waves enter the sensor array from one side. Consider a Broadside structure consisting of two microphones spaced 7.5 cm apart. The minimum response is observed when the signal is incident at an angle of 90° or 270° (in this case, the angle between the direction of the useful signal and the normal to the line of elements is taken as 0°). But this response strongly depends on the frequency of the received signal. Theoretically, such a system has a perfect zero at a frequency of 2.3 kHz. Above this frequency, depending on the direction of arrival, there are zeros at other angles (figure 3). The microphone array shows a clear directional characteristic at 4 kHz, and at 1 kHz its pattern is essentially omnidirectional. As a result, at lower frequencies the array cannot achieve significant spatial filtering.

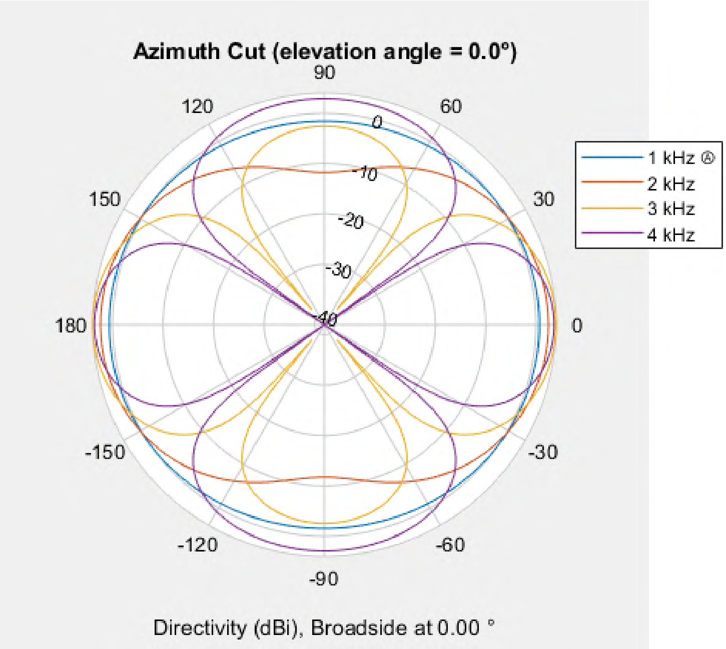


Figure 3: Dependence of signal reception on direction by a Broadside structure of two omnidirectional microphones for frequencies of 1 kHz, 2 kHz, 3 kHz and 4 kHz.

The Endfire structure consists of several microphones located in the direction of the useful acoustic signal. This design is called a differential array of microphones. The delayed signal from the first microphone is summed with the signal from the next microphone. To create a cardioid polar pattern, the signal from the rear microphones must be delayed by the same amount of time that the sound waves travel between the two microphone elements. Such structures are used to produce cardioid, hypercardioid or supercardioid directional response and theoretically completely eliminate sound incident on the array at an angle of 180°. A unidirectional microphone is more sensitive to sound coming from one direction and less sensitive to sounds from other directions. The most typical for such microphones is the cardioid characteristic, representing a peculiar diagram in the shape of a heart. At the same time, the peak of sensitivity is reached in the direction along the axis of the microphone, and the decline is in the opposite direction (figure 4).

To generate a cardioid response in direction, the signal from the omnidirectional microphones must be delayed for a time equal to the propagation of the acoustic wave between the two elements. Developers of such systems have two degrees of freedom to change the output signal of the speaker system: changing the distance between microphones and changing the delay time. Figure 5 shows the signal reception versus direction for various frequencies by the Endfire structure with two elements and a distance between them of 2.1 cm.

The distance between the microphones is crucial for the formation of a cardioid response. Figure 6 shows the same microphones, but placed at a distance of 15 cm.

The structures considered have the following advantages and disadvantages. Advantages of Broadside: flat geometry, simple processing implementation, ability to control the direction of the beam. Disadvantages of the Broadside: less off-axis rejection, close microphone spacing, and a large number of microphones needed to prevent spatial leakage.

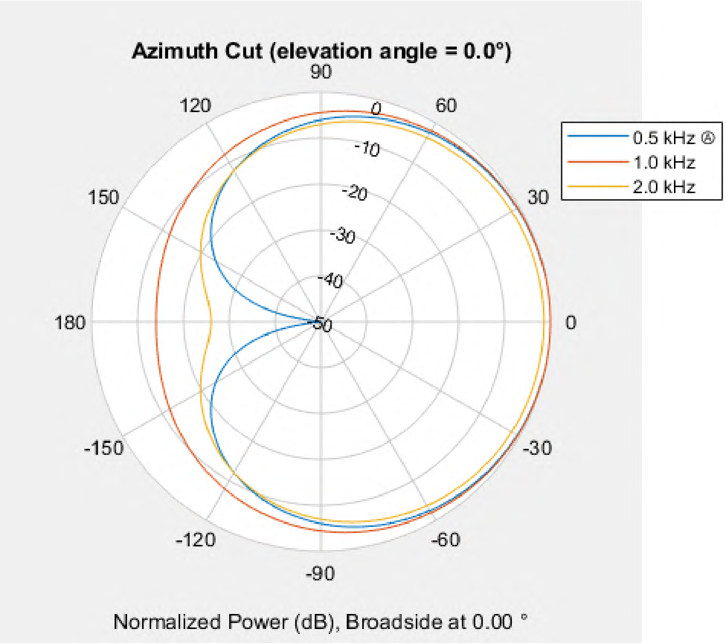


Figure 4: Dependence of signal reception on a unidirectional microphone for frequencies of 500 Hz, 1 kHz and 2 kHz.

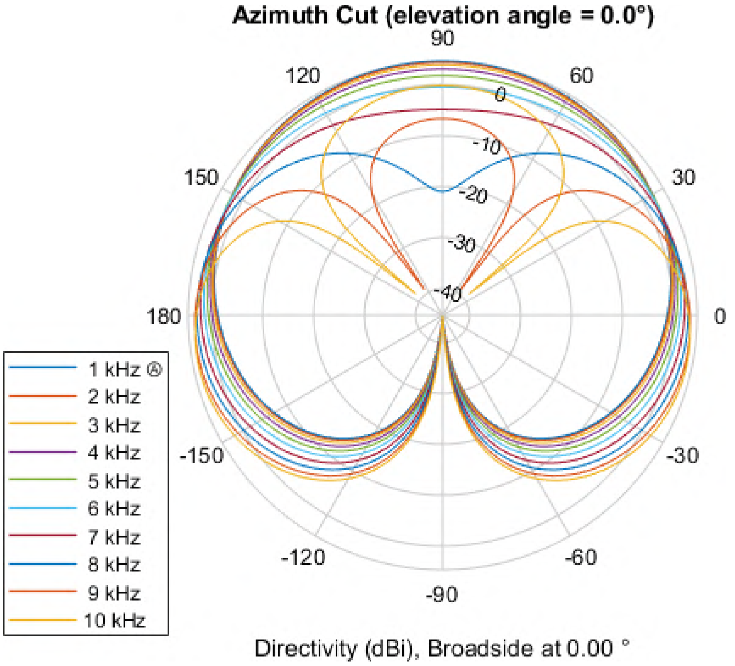


Figure 5: Dependence of signal reception on direction by an Endfire structure of two omnidirectional microphones for frequencies from 1 to 10 kHz, which are located at a distance of 21 cm.

Advantages of Endfire: Better off-axis suppression, smaller overall size. Disadvantages of Endfire: non-flat (volumetric) geometry, more complex processing, suppression of the useful signal in the low frequency range, the direction of the source of the useful signal must coincide with the axis of the microphone array; For two-dimensional gratings, beam formation is possible only in the horizontal direction (the grating array).

To form a differential array of higher orders, you need to add additional microphones. Since the petals will deviate more back and to the side in the directional diagram, the distance between the microphones will have to be increased. The figure 7 shows an array of 4 microphones (third order), which forms

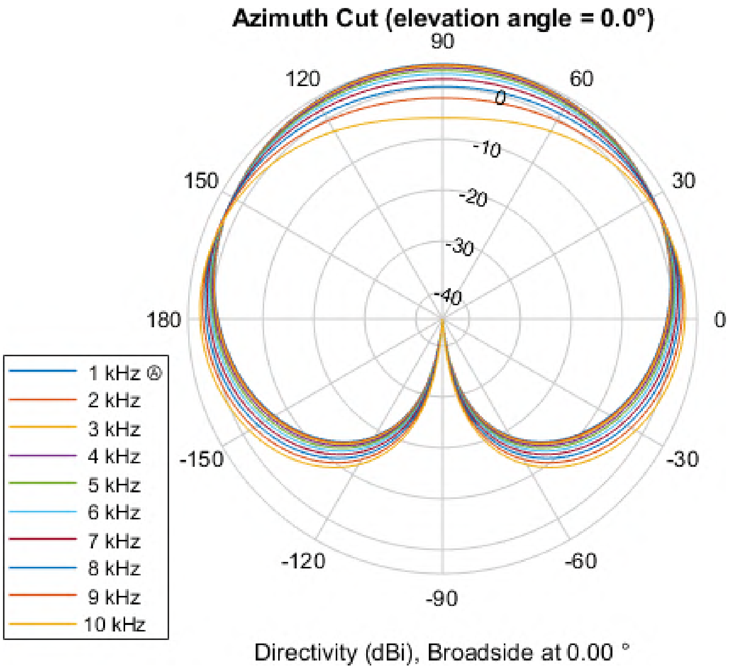


Figure 6: Dependence of signal reception on direction by an Endfire structure of two omnidirectional microphones for frequencies from 1 to 10 kHz, which are located at a distance of 15 cm.

a supercardioid pattern. Consider how beam formation depends on the number of microphones and the distance between them. It is worth noting that the sensitivity and frequency response of all array microphones must be precisely matched.

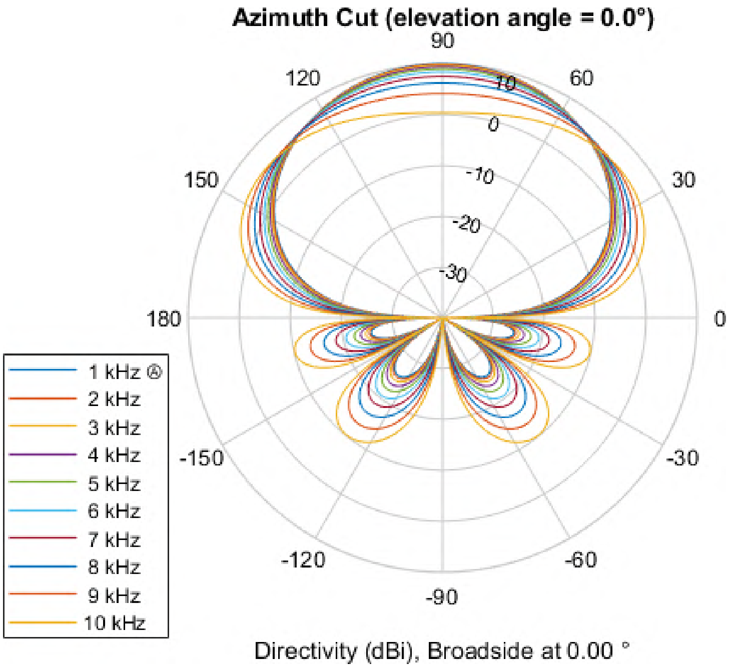


Figure 7: Differential array of 4 microphones (third order).

Differential microphone arrays make it possible to obtain high directivity characteristics of the system with its small size. But with such a construction, the problem arises of a significant change in the characteristics of the entire system with a slight deviation of the parameters of an individual microphone from its nominal values. If this approach is used for critical applications, measures must be taken to reduce deviations of microphone parameters from nominal values. As stated earlier, a first-order differential microphone array consists of two omnidirectional sensors separated by d (figure 8).

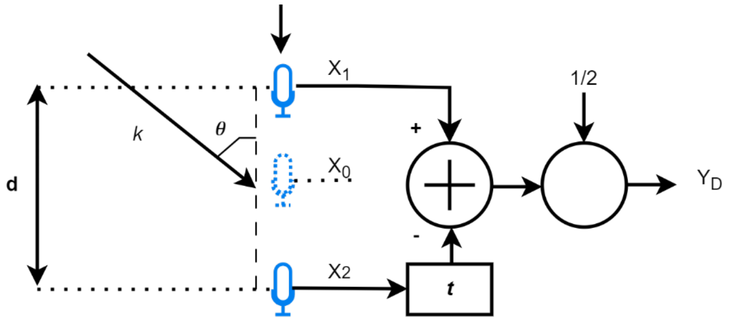


Figure 8: Structure of a first order differential microphone array.

4. Results

When sound arrives from the main direction $\theta = 0$, a delay appears between these sensors:

$$\tau_D = \frac{d}{c}, \tag{1}$$

where c is the speed of sound.

A plane wave, which is characterized by wave number $\overrightarrow{k}$, arrives at the input of the differential grating. Due to radial symmetry, the output signals of sensors $X_1(\omega)$ and $X_2(\omega)$ can be expressed by a function depending on the angle θ and frequency ω. There is a relationship $|\overrightarrow{k}|d = kd = \omega\tau_D$ between the wave number and the frequency of the signal. At the central point of the array, you can place a virtual microphone with an output signal $X_0(\omega)$. A plane wave incident at an angle θ with wave number causes $k = 2\pi/\lambda$ the appearance of signals at the output of microphones X_1 and X_2:

$$X_1(\omega) = X_0(\omega)e^{j\frac{kd}{2}\cos\theta},\ X_2(\omega) = X_0(\omega)e^{-j\frac{kd}{2}\cos\theta}. \tag{2}$$

At the output of the differential lattice we get

$$Y_D(\omega) = \frac{1}{2}(X_1(\omega) - X_2(\omega)e^{j\omega\tau}. \tag{3}$$

The directivity function of the differential array H_D is the ratio of the signal at the output of the array $Y_D(\omega)$ to the signal at the output of the virtual microphone $X_0(\omega)$:

$$H_D(\omega, \theta) = je^{-j\frac{\omega\tau}{2}} \sin\left(\frac{kd}{2}\left(\frac{\tau}{\tau_D} + \cos\theta\right)\right). \tag{4}$$

Usually very small values of $kd \ll 1$ are considered, which makes it possible to use the approximation $\sin\alpha \approx \alpha$. In this case, the idealized directivity function H_D has the form:

$$H_D(\omega, \theta) \approx \widetilde{H}_D(\theta) = j\frac{kd}{2}\left(\frac{\tau}{\tau_D} + \cos\theta\right). \tag{5}$$

With this view, the main characteristics of differential microphone arrays are obvious: 1) the form of $\widetilde{H}_D(\theta)$ is determined by the expression $\tau/\tau_D + \cos\theta$, which does not depend on frequency; 2) due to the subtraction of the signal a phase shift occurs by $\pi/2$; 3) the frequency response of the directivity function $H_D(\omega)$ has the form of a first-order high-pass filter.

At low frequencies, the output signal $Y_D(\omega)$ becomes highly susceptible to any changes in the shape of the characteristic $H_D(\omega)$. For this reason, the distance d should not be chosen too small, which may lead to a conflict with the condition $kd \ll 1$.

The exact expression for the directivity function (4) contains a sine function that scales the amplitude. It is rational to limit the operating range of the differential grating in the low frequency range to the first maximum of the sine. This first maximum fixes the cutoff frequency ωc:

$$\omega_c = \frac{\pi}{\tau_D + \tau}. \tag{6}$$

For low frequencies, the directivity characteristics are practically independent of frequency. However, as the frequency increases, the shape of the frequency response becomes more and more deformed. In addition, at some frequencies the signal is completely suppressed.

In order to compensate for the high-frequency nature of the behavior $H_D(\omega, \theta)$ it is necessary to develop a filter $W_{eq}(\omega)$. For the main direction $\theta = 0$, the adjusted frequency response$H_D(\omega, \theta = 0)W_{eq}(\omega)$ must be constant and equal to 0 dB, and for frequencies below ω_c:

$$W_{eq}(\omega) = \begin{cases} \frac{1}{\sin\left(\frac{\pi\omega}{2\omega_c}\right)}, & 0 < \omega < \omega_c, \\ 1, & \text{in other cases.} \end{cases} \tag{7}$$

For low frequencies $\omega \to 0$, the filter gain W_{eq} has very large values. This means that any noise present in the input signal will be greatly amplified. The level of this noise is determined by the specific sensor. This circumstance limits the frequency range of the signal for processing using a differential microphone array.

The directional properties of a microphone array are characterized by the directional coefficient (DI). It can be expressed as the ratio of the squared modulus of the directivity function in the main direction to the average value of the squared modulus in all directions:

$$DI(\omega) = \frac{|H(\omega, \theta = 0)|^2}{\frac{1}{4\pi}\int\limits_0^{2\pi}\int\limits_0^{\pi} |H(\omega, \theta)|^2 \sin\theta d\theta d\varphi}. \tag{8}$$

Taking into account the exact expression for the directivity function (4), we can obtain a new expression for the dependence of the directivity on frequency:

$$DI_D(\omega) = \frac{2sin^2\left(\frac{\omega}{2}\left(\tau_D + \tau\right)\right)}{1 - si\left(\omega\tau_D\right)\cos\left(\omega\tau\right)} \tag{9}$$

where $si(x) = \frac{1}{x}\sin(x)$.

The efficiency factor for low frequencies is obtained similarly to the result of approximation $\widetilde{H}_D$ according to expression (5):

$$\lim_{\omega \to 0} DI_D(\omega) = \widetilde{H}_D = \frac{3(\tau_D + \tau)^2}{3\tau_D^2 + 3\tau^2}. \tag{10}$$

Let us study the influence of microphone parameter mismatch for first-order differential arrays. We use a model of instability of microphone parameters in the form of a transfer function $M = M_{ref} + \Delta M$. The nominal transfer function of the sensor M_{ref} in this case is normalized to the value 1. It is assumed that the deviation ΔM is an independent random variable with variance:

$$\sigma_M^2 = E\{|\Delta M|^2\}, \tag{11}$$

where $E\{|\Delta M|^2\}$ expectation operator. Signals from two sensors in figure 8 will then be written as follows:

$$\hat{X}_1(\omega) = X_0(\omega)(1 + \Delta M_1)e^{j\frac{kd}{2}\cos\theta},\ \hat{X}_2(\omega) = X_0(\omega)(1 + \Delta M_2)e^{-j\frac{kd}{2}\cos\theta}. \tag{12}$$

The directivity function $\hat{H}_D$ for a differential array, taking into account the instability of the microphone parameters, can be obtained similarly to expression (4). But now there are additional conditions that depend on ΔM_i $(i = 1, 2)$. For random numbers, the quadratic terms remain, and the linear ones are set to zero, so we get:

$$E\{|\hat{H}_D(\omega,\theta)|^2\} = |H_D(\omega,\theta)|^2 + 2\sigma_M^2. \tag{13}$$

As a result, we can obtain a modified expression for DI:

$$E\{|\hat{DI}_D(\omega)|\} = \frac{2\sin^2\left(\frac{\omega}{2}\left(\tau_D+\tau\right)\right)+\sigma_M^2}{1-si\left(\omega\tau_D\right)\cos\left(\omega\tau\right)\sigma_M^2} \tag{14}$$

It is important to understand that in expression (13) the efficiency factor $H_D(\omega,\theta)$ characterizes the behavior of the system at high frequencies. While the W_{eq} equalization filter takes into account the effects of microphone instability and enhances them for low frequencies.

Thus, this work shows the dependence of the efficiency of a differential array of first-order microphones on frequency. It can be supplemented using a model of instability of microphone parameters at low frequencies.

Based on the presented dependence of the directivity on frequency and the instability model of the microphone parameters, a rational operating frequency range for the normal functioning of the microphone array can be determined. The lower limit of this range is limited by the instability of the microphone parameters, and the upper cutoff frequency is determined by the geometry of the array d.

Currently, there are many applications in which acoustic signals are processed. Microelectromechanical microphones (MEMS) are increasingly being used for these purposes. The use of such microphones allows the construction of differential microphone arrays. Microelectromechanical systems (MEMS) are a variety of microdevices of a wide variety of designs and purposes, in the production of which modified microelectronics technological techniques are used. Typically, all elements of such systems are placed on a common silicon base, the size of which is only a couple of millimeters. A MEMS microphone is an electro-acoustic device for converting sound vibrations into electrical waves, which is small enough to be installed in a tightly integrated product, for example: a smartphone, headset, speakerphone, laptop or any other device. There are two fundamentally important elements in such microphones: an integrated circuit (ASIC) and a MEMS sensor. It is the latter that ensures the capture and subsequent transmission of sound. The MEMS sensor itself consists of a flexible membrane and a rigidly fixed cover. Under the influence of air pressure, the membrane moves, changing the capacitance between the plates. This data is recalculated and output as an electrical signal to an integrated circuit. It is this signal that is converted into the sound that we hear.

Thanks to their design, MEMS microphones have the following advantages. Greater resistance to noise, vibration and temperature changes due to the absence of unnecessary connecting elements. Multiple MEMS microphones can be combined together to create a single array. Thanks to capacitive technology, these microphone arrays can capture sound from a precisely defined direction, effectively canceling echoes and background noise. Unlike other small microphones, such as electrets, MEMS microphones include more additional elements, such as preamplifiers, various filters and analog-to-digital converters. This means greater functionality while maintaining microscopic dimensions. Possibility of mounting such devices on the board using soldering.

Despite their many advantages, MEMS microphones are also not without their disadvantages. As we wrote above, MEMS microphones are often used as part of arrays, which increases the sound capture area, but at the same time reduces the service life of the devices. To work correctly, all microphones must work in unison, but the likelihood of one of them breaking is much higher than an individual device. Worse protection from moisture and dust than other microphones.

Microphone arrays include two or more built-in microphones, to which is added a programmable microprocessor designed to continuously determine the primary source of audio input and optimally adjust the output to achieve the best sound quality.

Let us highlight the most significant quality indicators of sound capture systems:

- useful signal/noise ratio, where the useful signal is the sound of the drone engine, and the noise is background noise, the microphone's own noise, and sounds from non-target sources;

- the shape of the radiation pattern and the ability of the system to change it depending on the environment;
- ability to localize the source of a useful signal and measurement accuracy parameters.

The most common way to build a signal capture unit is based on analog microphone arrays. A description of the problems that arise when developing analog microphone arrays, as well as the rationale for reducing the importance of the problem when using digital microphones in audio capture systems, is given in table 1.

Table 1
Problems encountered in the development of analog microphone arrays and justification for the feasibility of using digital microphones.

Problem when designing an analog array microphones	Rationale for using digital mirophones
Sharp increase in cost	Lack of a large number of auxiliary analog components
Reduced yield of suitable products due to the large number of components	Reducing the total number of microcircuits and topological complexity leads to an increase in the percentage of usable products due to the general laws of statistics
Increased development and debugging costs	Implementation of algorithms in code and digital interface blocks, which allows you to attract developers with less qualifications and experience
High sensitivity to electromagnetic radiation and power quality	The use of digital components is less sensitive to static failures and degradation of power supply quality
Increased production cycle	Less topological complexity guarantees the ability to produce a product according to almost any modern technological standards, making the launch process faster and cheaper
Increasing the testing cycle	The digital implementation allows you to write synthetic tests and generate input signals in the same way. Digital generators are more flexible and low cost, and the testing and debugging process is reduced to working with code

As an alternative to existing approaches that have the disadvantages outlined above, the authors of this work proposed to use an architecture built using digital MEMS microphones, which have become widespread recently. The meter for these microphones is located on-chip, so its digital output is minimally affected by the components that surround it. The most simple, inexpensive and perfect solution in terms of signal capture parameters was developed, which consists of using digital MEMS microphones and an Arduino microcomputer.

When choosing a digital microphone for use in a linear differential microphone array, it is important to consider the following factors:

- Sensitivity: The sensitivity of the microphone is a measure of how well it can convert sound waves into electrical signals. A higher sensitivity microphone will be able to pick up quieter sounds, but it may also be more susceptible to noise.
- Signal-to-Noise Ratio (SNR): The SNR of the microphone is a measure of the ratio of the desired signal (sound) to the undesired signal (noise). A higher SNR microphone will have less noise, resulting in cleaner recordings.
- Dynamic range: The dynamic range of the microphone is the range of sound pressure levels that it can accurately measure. A wider dynamic range microphone will be able to capture both very loud and very quiet sounds without distortion.
- Linearity: The linearity of the microphone is a measure of how accurately it can reproduce the input signal. A more linear microphone will produce recordings that are more faithful to the original sound. In addition to the above factors, it is also important to consider the cost and availability of the microphone when making a selection.

Each digital MEMS microphone can be simplified into the model shown in figure 9. Input sound vibrations are converted through a MEMS membrane into a weak electrical signal, which is then fed to the input of amplifier A. The pre-amplified signal then passes through an analog low-pass filter (LPF), which is necessary to protect against aliasing. The final element of signal processing in the microphone is a 4th order $\Sigma - \Delta$ modulator, which converts the input analog signal into a one-bit digital stream. The frequency of data bits from the output of the $\Sigma - \Delta$ modulator is equal to the frequency of the input timing signal CLK and, as a rule, lies in the range from 1 to 4 MHz.

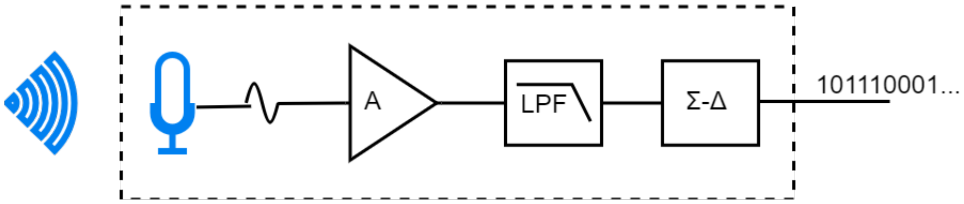


Figure 9: A simple model of a digital MEMS microphone.

In the time domain, the output of a $\Sigma - \Delta$ modulator is a jumbled collection of ones and zeros. However, if we assign a value of 1.0 to each high logical level of the microphone output, and a value of −1.0 to each low level and then perform a Fourier transform, we will obtain a spectrogram of the output data from the microphone.

Let’s look at the pins of a digital microphone. VDD – microphone power supply, GND – Ground, CLK – input clock signal, synchronously with which the DATA line switches its DATA states. During one half of the CLK cycle this pin is in a high impedance state, and during the second half it serves as a pin for reading data from the $\Sigma - \Delta$ output of the microphone modulator. L/R_{Sel} – this pin is used to control switching of the DATA line. If L/R_{Sel} is connected to VDD, then after some time after detecting the rising edge of the CLK signal, the DATA pin goes into a high impedance state, and after the arrival of the falling edge of the CLK signal, the DATA pin is connected to the $\Sigma - \Delta$ output of the microphone modulator. If L/R_{Sel} is connected to GND, the edges of the CLK signal, along which the DATA line switches, are reversed (figure 10).

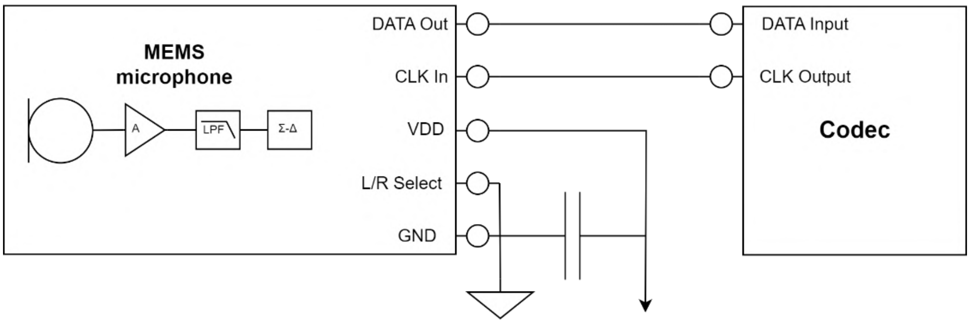


Figure 10: The pins of a digital microphone.

To isolate the audio frequency band signal, the data from the microphone must be filtered and resampled at a lower frequency (usually 50–128 times lower than the sampling frequency of the $\Sigma - \Delta$ modulator). A digital low-pass filter filters out external noise and the microphone’s own noise outside the operating band $(f > F_{CLK}/2M)$ to protect against aliasing, and also makes it possible to reduce the data repetition rate. In figure 11 presents one of the possible options for processing a one-bit data stream from a microphone, implemented in software on a DSP or in hardware in audio codecs. Shown in figure 11, the sampling frequency compression circuit (compressor) lowers the sampling frequency due to the fact that from every M samples of the filtered signal $w(mM)$, $M\breve{}1$ sample is discarded. The input and output of the converter shown in figure 8 are related by the following expression:

$$y(m) = w(mM) = \sum_{k=-\infty}^{\infty} h(k)x(mM-k). \tag{15}$$

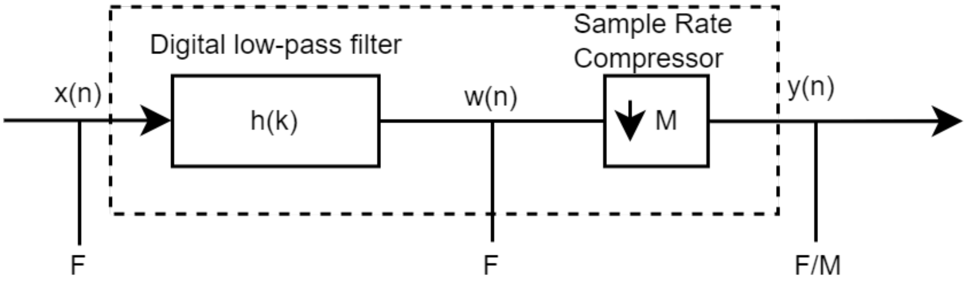


Figure 11: Signal conversion by $\Sigma - \Delta$ modulator.

MEMS microphones have PDM (Pulse-Density Modulation, PDM) outputs. Pulse density modulation is a method of transmitting the relative change in signal per sample, which can be mathematically described by the formula

$$x[n] = -A(-1^{a[n]}), \tag{16}$$

where $x[n]$ contains in each term the relative change in the signal in the form of 1 bit with a sign, which is specified by the transition. A negative increment is a transition from 1 to 0, a positive increment is from 0 to 1. Repeating ones increases the overall amplitude of the signal, and repeating zeros decreases (figure 12).

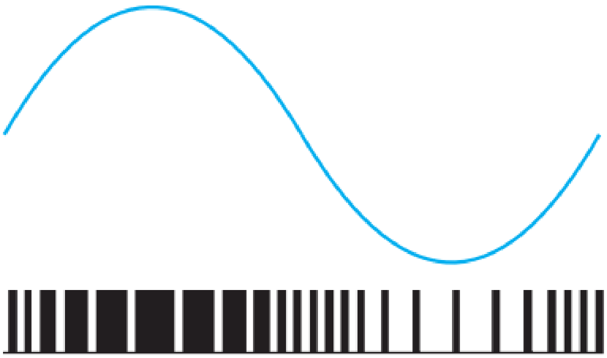


Figure 12: Period of a sine wave per 100 samples.

A mathematical model for pulse density modulation can be obtained using a delta-sigma modulator model. In the discrete frequency domain, the operation of a delta-sigma modulator can be described by the formula

$$O(z) = I(z) + E(z)(1 - z^{-1}), \tag{17}$$

where $O(z)$, $I(z)$ are the signal spectra at the input and output of the modulator; $E(z)$ is the sampling error of the delta-sigma modulator; $1 - z^{-1}$ is high-pass filter. As a result of transforming the formula, we get

$$O(z) = E(z)[I(z) - O(z)z^{-1})]\frac{1}{1-z^{-1}}. \tag{18}$$

According to this formula, the error $E(z)$ reduces the value of the signal at the output $O(z)$ in the low-frequency region and increases it in the high-frequency region, as a result of which the quantization noise spectrum shifts predominantly to the high-frequency region.

Table 2
Array characteristics (ULA with 2 microphones).

Array characteristic	Value
Array directivity	2.97 dBi at 0 Az; 0 El
Array span	x=0 m y=17 mm z=0 m
Number of elements	2
HPBW	60.50° Az / 360.00° El
FNBW	180.00° Az / -° El
SLL	- dB Az / - dB El
Element polarization	None

Let $i[n]$ be a sample of the signal at the input of the modulator in the time domain, and $o[n]$ be a sample of the output signal, then, using the inverse z-transform, we can proceed to the expression

$$o[n] = i[n] + e[n] - e[n-1], \tag{19}$$

where

$$o[n] = \begin{cases} 1 & \text{if} x[n] \geq e[n-1]; \\ -1 & \text{if} x[n] < e[n-1]; \end{cases} \tag{20}$$

$$e[n] = o[n] - i[n] + e[n-1]. \tag{21}$$

The signal from the output signal sample $o[n]$ is represented as 1 bit and takes values ± 1, and is implemented so that the value of the current quantization error $e[n]$ is minimal. In this case, the quantization error $e[n]$ of each sample appears at the device input during the subsequent sample.

When implementing frequency converters in software, a finite impulse response (FIR) filter or an Infinite impulse response (IIR) filter can be used as a digital LPF. Developers should be very careful when choosing the type of filter, its length and bit depth, since the performance of the entire system as a whole directly depends on this. A correctly calculated and implemented decimator (frequency converter) in some cases will significantly reduce the cost of products and increase its technical characteristics.

As a second option, audio codecs adapted for this can be used to convert data from the output of a digital microphone, which will significantly reduce product development time. For example, Analog Devices offers the ADAU1361 and ADAU1761 codecs, which are suitable for the ADMP521 microphones. In our work we used a microphone ADMP521. However, the process of creating digital audio devices becomes simple in terms of hardware implementation and complex in terms of writing programs for the microcontrollers used.

Next, we conducted a simulation and computational experiment of a uniform linear array of 2 omnidirectional microphones using Matlab Sensor Array Analyzer.

The following model parameters were used. The distance between the microphones is 20 mm. The board has two MEMS microphones spaced 20 mm apart. This spacing is ideal for detecting acoustic events. Additionally, the 20 mm spacing is equivalent to $8 \cdot 2.54$ mm, which makes it suitable for DIP (Dual In-line Package) – a type of housing for microcircuits, electronic modules and some other electronic components. Experimentally, a distance of 0.017 m was determined for the formation of a bi-directional pattern.

Next, we conducted a simulation and computational experiment of a uniform linear array of 2 omnidirectional microphones using Matlab Sensor Array Analyzer. The following model parameters were used. The distance between the microphones is 17 mm. The speed of sound is 343 m/s, the signal frequency is 10 kHz. As a result, we obtained the parameters listed in table 2.

The Matlab script is listed below:

```
% Create a Uniform Linear Array Object
Array = phased.ULA('NumElements',2, 'ArrayAxis','y');
Array.ElementSpacing = 0.017;
Array.Taper = ones(1,2).';
% Create an omnidirectional microphone element
Elem = phased.OmnidirectionalMicrophoneElement;
Elem.FrequencyRange = [0 10000];
Array.Element = Elem;
% Assign Frequencies and Propagation Speed
Frequency = 10000;
PropagationSpeed = 343;
% Create Figure
% Plot Array Geometry figure;
viewArray(Array,'ShowNormal',true,
'ShowTaper',true,'ShowIndex','All',
'ShowLocalCoordinates',true,'ShowAnnotation',true,
'Orientation',[0;0;0]);
% Find the weights
w = ones(getNumElements(Array), length(Frequency));
% Plot 2d azimuth graph
format = 'polar';
cutAngle = 0;
plotType = 'Directivity';
plotStyle = 'Overlay';
figure;
pattern(Array, Frequency, -180:180, cutAngle, 'PropagationSpeed',
PropagationSpeed, 'CoordinateSystem', format ,'weights', w,
'Type', plotType, 'PlotStyle', plotStyle);
% Find the weights
w = ones(getNumElements(Array), length(Frequency));
% Plot 2d elevation graph
format = 'polar';
cutAngle = 0;
plotType = 'Directivity';
plotStyle = 'Overlay';
figure;
pattern(Array, Frequency, cutAngle, -90:90, 'PropagationSpeed',
PropagationSpeed, 'CoordinateSystem', format ,'weights', w,
'Type', plotType, 'PlotStyle', plotStyle);
% Find the weights
w = ones(getNumElements(Array), length(Frequency));
% Plot U Pattern
format = 'uv';
plotType = 'Directivity';
plotStyle = 'Overlay';
figure;
pattern(Array, Frequency, -1:0.01:1, 0, 'PropagationSpeed',
PropagationSpeed, 'CoordinateSystem', format,'weights', w,
'Type', plotType, 'PlotStyle', plotStyle);
```

The resulting pattern has the shape of a bi-directional (figure 13). As you can see, the design of the grille allows you to create a grille with the main lobes directed at -90 and 90 degrees. To form a cardiode radiation pattern, as mentioned above, it is necessary to use delay-and-sum and filter-and-sum algorithms. The meaning of these algorithms is that microphone signals are added with different delays (different phase shifts), aligning the phases of signals coming from the selected direction (source localization) for each frequency. In this case, the beamforming algorithm makes it possible to amplify the signals generated by sound coming from the selected direction, i.e. performs a kind of focusing of sounds.

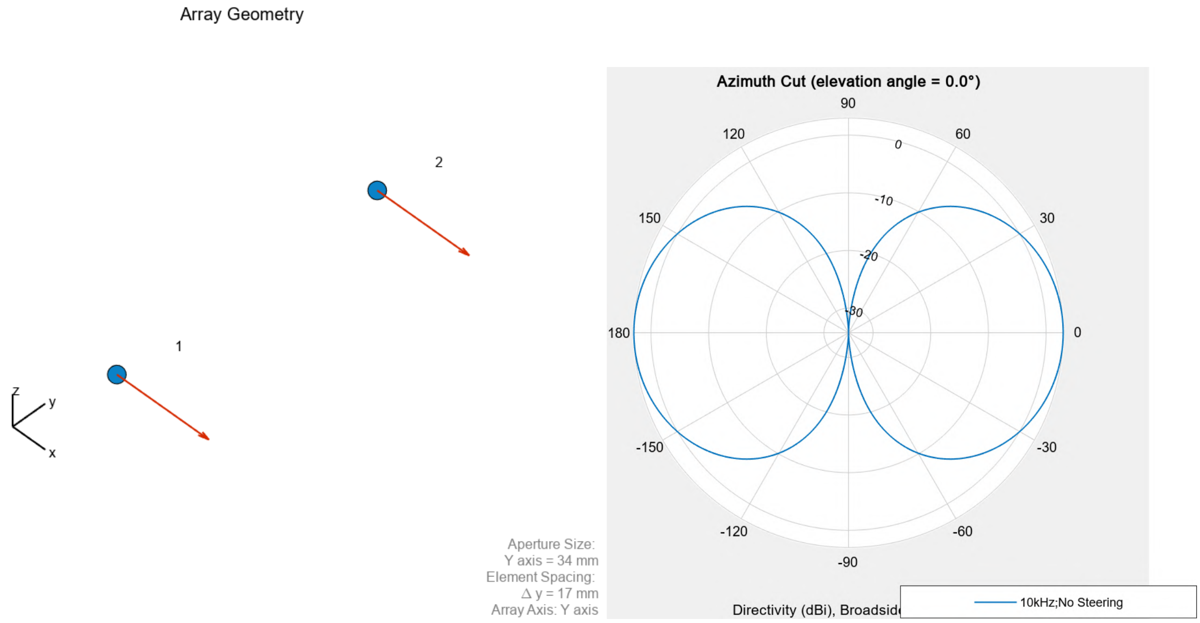


Figure 13: The geometry of the array and the directional diagram (linear array of 2 microphones).

ADMP521 microphones were connected to the ADAU1761 codec in accordance with the technical specifications of both products (figure 14).

A model was also created based on an array of four omnidirectional microphones located at a distance of 20 mm (figure 15). The following model parameters were used. The distance between the microphones is 17 mm. The speed of sound is 343 m/s, the signal frequency is 10 kHz. As a result, we obtained the parameters listed in table 3.

Table 3
Array characteristics (ULA with 4 microphones).

Array characteristic	Value
Array directivity	5.98 dBi at 0 Az; 0 El
Array span	x=0 m y=51 mm z=0 m
Number of elements	4
HPBW	26.52° Az / 360.00° El
FNBW	60.58° Az / -° El
SLL	11.30 dB Az / - dB El
Element Polarization	None

The Matlab script has the following form:

```
% Create a Uniform Linear Array Object
Array = phased.ULA('NumElements',4, 'ArrayAxis','y');
Array.ElementSpacing = 0.017;
Array.Taper = ones(1,4).';
% Create an omnidirectional microphone element
Elem = phased.OmnidirectionalMicrophoneElement;
Elem.FrequencyRange = [0 10000];
Array.Element = Elem;
```

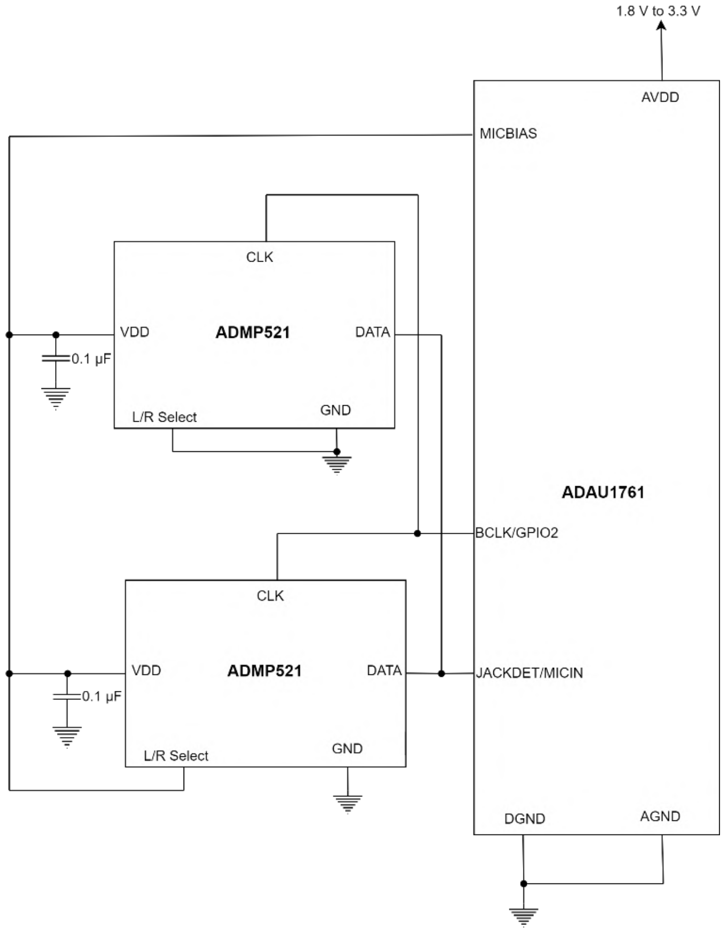


Figure 14: Connection diagram of ADMP521 microphones to ADAU1761 codec.

```
% Assign Frequencies and Propagation Speed
Frequency = 10000;
PropagationSpeed = 343;
% Create Figure
% Plot Array Geometry
figure;
viewArray(Array,’ShowNormal’,true,’ShowTaper’,true,’ShowIndex’,’All’,
’ShowLocalCoordinates’,true,’ShowAnnotation’,true,’Orientation’,[0;0;0]);
% Calculate Steering Weights
Freq3D = 10000;
% Find the weights
w = ones(getNumElements(Array), length(Frequency));
% Plot 3d graph
format = ’polar’;
plotType = ’Directivity’;
figure;
pattern(Array, Freq3D , ’PropagationSpeed’, PropagationSpeed,
’CoordinateSystem’, format,’weights’, w(:,1),
’ShowArray’,false,’ShowLocalCoordinates’,true,
’ShowColorbar’,true,’Orientation’,[0;0;0],’Type’, plotType);
% Find the weights
w = ones(getNumElements(Array), length(Frequency));
```

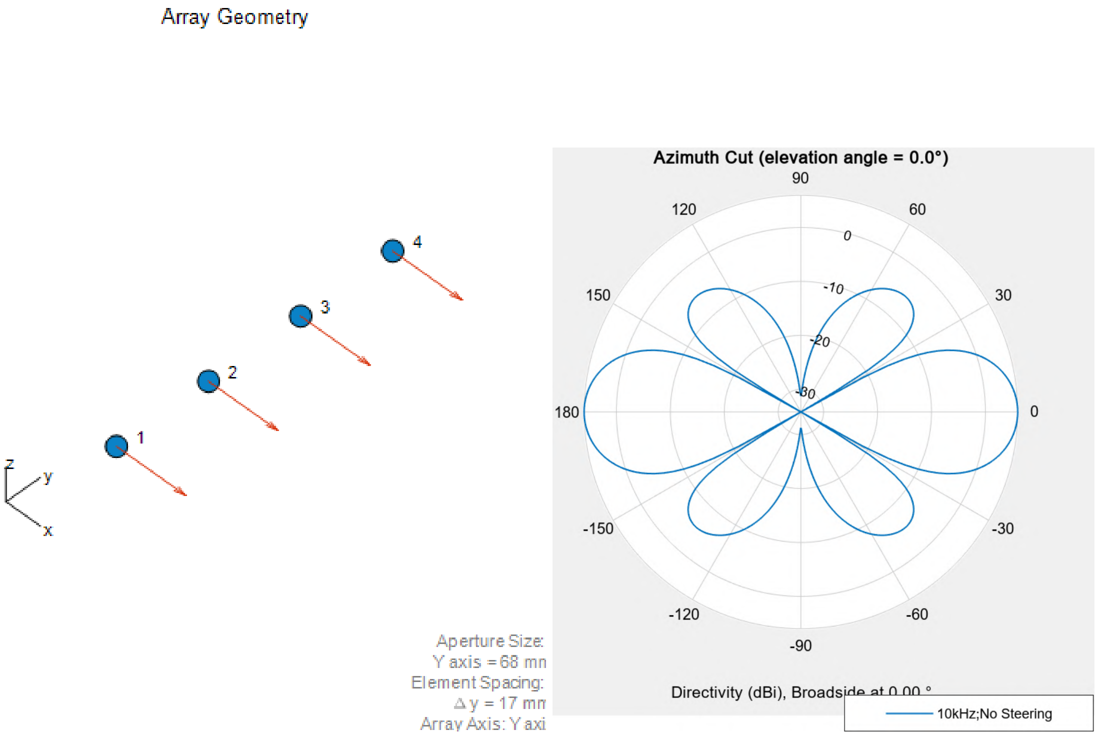


Figure 15: The geometry of the array and the directional diagram (linear array of 4 microphones).

```
% Plot 2d azimuth graph
format = 'polar';
cutAngle = 0;
plotType = 'Directivity';
plotStyle = 'Overlay';
figure;
pattern(Array, Frequency, -180:180, cutAngle, 'PropagationSpeed',
PropagationSpeed,'CoordinateSystem', format ,'weights', w,
'Type', plotType, 'PlotStyle', plotStyle);
% Find the weights
w = ones(getNumElements(Array), length(Frequency));
% Plot 2d elevation graph
format = 'polar';
cutAngle = 0;
plotType = 'Directivity';
plotStyle = 'Overlay';
figure;
pattern(Array, Frequency, cutAngle, -90:90, 'PropagationSpeed',
PropagationSpeed,'CoordinateSystem', format ,'weights', w,
'Type', plotType, 'PlotStyle', plotStyle);
```

So, during the computational experiment, we built 2 linear microphone arrays with bi-directionality. The directionality of these arrays can be easily converted to unidirectional (cardioid) using known algorithms or hardware (codecs). The tuning of the circuit to create cardioid directivity will be considered in further studies.

5. Discussion

The following questions require additional discussion and clarification: the dependence of the azimuthal pattern of the proposed linear microphone arrays on the source frequency; the choice of analog or digital MEMS microphones and labor-intensiveness in the development of a microphone array; the use of directional microphones instead of omnidirectional; peculiarities of localization of a moving sound source (Doppler effect, reflection from obstacles, etc.); higher-order differential beam array formers; signal processing algorithms of microphone arrays.

6. Conclusions

The study looks at setting up a microphone array to determine the position of a UAV (unmanned aerial vehicle) based solely on the sound of its engines. The location of the microphones plays a crucial role for accurate localization. A mathematical model of pulse density modulation of a digital MEMS microphone is also considered. This work shows the dependence of the efficiency of a differential array of first-order microphones on frequency. Based on the presented dependence of the directivity on frequency and the instability model of the microphone parameters, a rational operating frequency range for the normal functioning of the microphone array can be determined.

A model of a linear microphone array based on MEMS omnidirectional microphones is proposed, which with a certain geometrical arrangement give a bi-directional pattern, which, in principle, can be easily transformed into a unidirectional one with the use of special algorithms or hardware (for example, ADAU1761 codecs). Refinement of the circuit to achieve cardioid directivity will be addressed in forthcoming research.

7. Author contributions

Conceptualization, methodology – Andrii V. Riabko, Oksana V. Zaika; setting tasks, conceptual analysis – Tetiana A. Vakaliuk, Oksana V. Zaika; development of the model – Andrii V. Riabko, Valerii V. Kontsedailo; software development, verification – Andrii V. Riabko, Roman P. Kukharchuk; analysis of results, visualization – Roman P. Kukharchuk, Tetiana A. Vakaliuk; drafting of the manuscript – Valerii V. Kontsedailo, reviewing and editing – Tetiana A. Vakaliuk.

All authors have read and approved the published version of this manuscript.

References

[1] M. Cobos, F. Antonacci, A. Alexandridis, A. Mouchtaris, B. Lee, A Survey of Sound Source Localization Methods in Wireless Acoustic Sensor Networks, Wireless Communications and Mobile Computing 2017 (2017) 3956282. doi:10.1155/2017/3956282.

[2] A. R. Petrosian, R. V. Petrosyan, I. A. Pilkevych, M. S. Graf, Efficient model of PID controller of unmanned aerial vehicle, Journal of Edge Computing 2 (2023) 104–124. doi:10.55056/jec.593.

[3] M. Risoud, J.-N. Hanson, F. Gauvrit, C. Renard, P.-E. Lemesre, N.-X. Bonne, C. Vincent, Sound source localization, European Annals of Otorhinolaryngology, Head and Neck Diseases 135 (2018) 259–264. doi:10.1016/j.anorl.2018.04.009.

[4] W. A. Yost, M. T. Pastore, Y. Zhou, Sound Source Localization Is a Multisystem Process, Springer International Publishing, Cham, 2021, pp. 47–79. doi:10.1007/978-3-030-57100-9_3.

[5] E. King, A. Tatoglu, D. Iglesias, A. Matriss, Audio-visual based non-line-of-sight sound source localization: A feasibility study, Applied Acoustics 171 (2021) 107674. doi:10.1016/j.apacoust.2020.107674.

[6] P. Chiariotti, M. Martarelli, P. Castellini, Acoustic beamforming for noise source localization – reviews, methodology and applications, Mechanical Systems and Signal Processing 120 (2019) 422–448. doi:10.1016/j.ymssp.2018.09.019.

[7] T. Yamada, K. Itoyama, K. Nishida, K. Nakadai, Sound Source Tracking by Drones with Microphone Arrays, in: 2020 IEEE/SICE International Symposium on System Integration (SII), 2020, pp. 796–801. doi:10.1109/SII46433.2020.9026185.

[8] A. D. Firoozabadi, P. Irarrazaval, P. Adasme, D. Zabala-Blanco, P. Palacios-Játiva, H. Durney, M. Sanhueza, C. Azurdia-Meza, Three-dimensional sound source localization by distributed microphone arrays, in: 2021 29th European Signal Processing Conference (EUSIPCO), 2021, pp. 196–200. doi:10.23919/EUSIPCO54536.2021.9616326.

[9] Y. Sasaki, R. Tanabe, H. Takemura, Probabilistic 3D sound source mapping using moving microphone array, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 1293–1298. doi:10.1109/IROS.2016.7759214.

[10] M. C. Catalbas, M. Yildirim, A. Gulten, H. Kurum, S. Dobrišek, Estimation of Trajectory and Location for Mobile Sound Source, International Journal of Advanced Computer Science and Applications 7 (2016). doi:10.14569/IJACSA.2016.070934.

[11] K. Hoshiba, K. Washizaki, M. Wakabayashi, T. Ishiki, M. Kumon, Y. Bando, D. Gabriel, K. Nakadai, H. G. Okuno, Design of UAV-Embedded Microphone Array System for Sound Source Localization in Outdoor Environments, Sensors 17 (2017) 2535. doi:10.3390/s17112535.

[12] T. Tachikawa, K. Yatabe, Y. Oikawa, 3D sound source localization based on coherence-adjusted monopole dictionary and modified convex clustering, Applied Acoustics 139 (2018) 267–281. doi:10.1016/j.apacoust.2018.04.033.

[13] M. Wakabayashi, H. G. Okuno, M. Kumon, Drone audition listening from the sky estimates multiple sound source positions by integrating sound source localization and data association, Advanced Robotics 34 (2020) 744–755. doi:10.1080/01691864.2020.1757506.

[14] L. Carneiro, A. Berry, Three-dimensional sound source diagnostic using a spherical microphone array from multiple capture positions, Mechanical Systems and Signal Processing 199 (2023) 110455. doi:10.1016/j.ymssp.2023.110455.

[15] S. Kita, Y. Kajikawa, Fundamental study on sound source localization inside a structure using a deep neural network and computer-aided engineering, Journal of Sound and Vibration 513 (2021) 116400. doi:10.1016/j.jsv.2021.116400.

[16] F. R. do Amaral, J. Rico, M. A. F. de Medeiros, Design of microphone phased arrays for acoustic beamforming, Journal of the Brazilian Society of Mechanical Sciences and Engineering 40 (2018) 354. doi:10.1007/s40430-018-1275-5.

[17] F. Grondin, F. Michaud, Lightweight and optimized sound source localization and tracking methods for open and closed microphone array configurations, Robotics and Autonomous Systems 113 (2019) 63–80. doi:10.1016/j.robot.2019.01.002.

[18] C. Rascon, I. Meza, Localization of sound sources in robotics: A review, Robotics and Autonomous Systems 96 (2017) 184–210. doi:10.1016/j.robot.2017.07.011.

[19] M. J. Bianco, P. Gerstoft, J. Traer, E. Ozanich, M. A. Roch, S. Gannot, C.-A. Deledalle, Machine learning in acoustics: Theory and applications, The Journal of the Acoustical Society of America 146 (2019) 3590–3628. doi:10.1121/1.5133944.

[20] H. Niu, Z. Gong, E. Ozanich, P. Gerstoft, H. Wang, Z. Li, Deep-learning source localization using multi-frequency magnitude-only data, The Journal of the Acoustical Society of America 146 (2019) 211–222. doi:10.1121/1.5116016.

[21] W. He, P. Motlicek, J.-M. Odobez, Deep Neural Networks for Multiple Speaker Detection and Localization, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018, pp. 74–79. doi:10.1109/ICRA.2018.8461267.

[22] A. Ebrahimkhanlou, S. Salamone, Single-Sensor Acoustic Emission Source Localization in Plate-Like Structures Using Deep Learning, Aerospace 5 (2018) 50. doi:10.3390/aerospace5020050.

[23] S. Adavanne, A. Politis, J. Nikunen, T. Virtanen, Sound Event Localization and Detection of Overlapping Sources Using Convolutional Recurrent Neural Networks, IEEE Journal of Selected Topics in Signal Processing 13 (2019) 34–48. doi:10.1109/JSTSP.2018.2885636.

[24] G. Chardon, Theoretical analysis of beamforming steering vector formulations for acoustic source localization, Journal of Sound and Vibration 517 (2022) 116544. doi:10.1016/j.jsv.2021.

116544.

[25] B. da Silva, A. Braeken, K. Steenhaut, A. Touhafi, Design Considerations When Accelerating an FPGA-Based Digital Microphone Array for Sound-Source Localization, J. Sensors 2017 (2017) 6782176:1–6782176:20. doi:10.1155/2017/6782176.

Search and classification of objects in the zone of reservoirs and coastal zones

Viktorija M. Smolij[1], Natan V. Smolij[2] and Sergii P. Sayapin[1]

[1]National University of Life and Environmental Sciences of Ukraine, 15 Heroyiv Oborony Str., Kyiv, 03041, Ukraine
[2]Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37 Beresteiskyi Ave., Kyiv, 03056, Ukraine

Abstract

A minimal working version of the computer vision subsystem has been developed specifically for deployment on a research unmanned aerial vehicle (UAV). This subsystem focuses on detecting specific objects present on the surfaces of water bodies and subsequently classifying them. The effectiveness of this subsystem was evaluated by comparing two state-of-the-art models, YOLOv5 and YOLOv8, to determine their suitability for addressing the target problem. To evaluate performance of the resulted model's series of test was performed. It resulted in achieving desired output of object detections but with low accuracy of classification, however such systems can be used as wider-area object detector. According to the obtained results, it can be seen that the system detects objects on the water surface, but the classification of these objects is not good. There are several reasons for this: errors in the labeling of the dataset and the small size of the dataset. A possible scenario of using the built model is the general collection of information about the reservoir without regard to the classification output. In the process of such exploitation, it can be considered as expedient to collect a dataset that will correspond to the data from the drone (the data of the current dataset is data from surveillance cameras and video recordings from boats). In the future, form the dataset according to the developer's requirements, applying the necessary data augmentation steps.

Keywords

dataset, model, image distribution, confusion matrix, training metrics, augmentation, mosaic placement of images

1. Introduction

In the modern world of robotics, many tasks require the intervention of artificial intelligence to increase the number of tasks to be solved [1, 2, 3, 4], increase productivity, reduce execution time, scale processing, exclude a person from the process of performing routine tasks, and ensure online information collection and processing processes [5, 6, 7, 8]. So, for example, creating maps and patrolling water bodies using traditional methods is a time-consuming and time-consuming process that can be improved and accelerated with the help of artificial intelligence [9, 10, 11].

In the modern period of development of unmanned aerial vehicles comes the realization that many tasks of research and observation can be transferred to automated drones, which will perform them faster and better due to the possibility of installing additional computing power as a payload [12, 13, 14, 15]. This approach is also supported by the fact that flight controllers available on the market, such as Betaflight, Pixhawk, etc. [16, 17], have a wide range of interfaces for communicating with external devices, exchanging telemetry information, camera data and other interesting data sets that can be grouped into datasets for automation management and debugging processes [18, 19].

Computer vision systems are technologies that give computers the ability to recognize and analyze visual data [20]. The structure of a computer vision system is usually complex and depends on the specific task and the technologies used [21, 22]. However, generally speaking, a computer vision system can be divided into several key components: data collection, data pre-processing, feature summarization, recognition and classification, decision-making process, presentation of results, etc.

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine

vmsmolij@nubip.edu.ua (V. M. Smolij); hoibbitizukrainy@gmail.com (N. V. Smolij); sayapin@nubip.edu.ua (S. P. Sayapin)
https://nubip.edu.ua/node/3062 (V. M. Smolij); https://www.facebook.com/profile.php?id=61553465353363 (N. V. Smolij); https://nubip.edu.ua/node/3062 (S. P. Sayapin)
0000-0002-1268-7837 (V. M. Smolij); 0009-0002-3763-6726 (N. V. Smolij); 0000-0003-1565-4034 (S. P. Sayapin)

Unmanned aerial vehicles (UAVs) have various structural elements that determine their functionality and characteristics. The main structural elements of a UAV include: fuselage (body), wings, tail unit, engines, connecting elements, equipment for filming and observation (cameras, sensors). The variety of tasks for the use of UAVs in water and coastal zones illustrate the importance and relevance of the conducted research for various fields of application, including full-scale war on the territory of Ukraine, customs, security, monitoring, ecology and natural science. The purpose of the research is to create a minimum working version of the computer vision subsystem for use on a research UAV and to provide instructions for further improvement of the system and its development.

The feasibility of using AI is due to the fact that there is no clear algorithm for detecting objects on the water surface using image processing methods other than AI. Also, the use of UAVs in combination with AI will allow processing data from large areas of the earth and water surface, which will improve the response to emergency situations with the use of a limited number of human resources [23].

For task of object detection in the image, there are a large number of software solutions that allow you to construct and train a neural network. Examples of such solutions are tensorflow, pytorch, theano, ultralytics, chainer libraries. Since the task of creating a dataset is part of the usual functionality of the libraries, the range of possible options is narrowed to the ultralytics API, which is less flexible in terms of model selection, but provides a wide functionality for working with data. To perform the given task, it is most appropriate to use the Ultralytics API, as they provide the necessary functionality for dataset synthesis and provide interfaces for programming the training of a wide range of models for object detection. The software is written in the Python programming language due to its dynamic typing and automatic garbage collection, as well as a port of the above API for this language.

2. A model of an artificial intelligence system

2.1. Creating a dataset

The subsystem will control the drone, which must move along the route at the points specified by the user, and be able to detect such objects as boats, ships, buoys, garbage islands, swimmers and drowning people from the image from the camera.

The dataset is under development and is a compilation from several data sources: https://universe.roboflow.com/hamdi-ali/plastic-pollution-ugslg, https://universe.roboflow.com/double-o-co-ltd/marine-object-detection-yjybm/dataset/4, https://universe.roboflow.com/pwnface4-gmail-com/drowning-people. The general principles, application, versatility and features of use are given in [24, 25, 26, 27, 28]

Model definition: since it is planned to implement the model on a Raspberry microcomputer, 2 possible models for training can be distinguished, YOLOv5 due to its small size and YOLOv8 due to the fact that with approximately the same number of parameters as YOLOv5, the model gives a better result, as shown in figure 1.

In the figure 1, the number of parameters is shown on the abscissa axis, and the effectiveness is shown on the ordinate axis.

The following are examples of the considered images. The “Boat” object class is shown in figure 3.

The object class “Ship” is shown in figure 4.

The object class “Buoy” is given on figure 5.

The class of objects “Swimmer” is shown in figure 6.

The object class “Drowning man” is shown in figure 7.

The distribution of images by classes is shown in figure 8.

The largest number of markings belongs to the boat class (1629 units). Along with this class, the following classes are well represented (in descending order): swimmer, boat and buoy, respectively 1498, 1270 and 1186. The garbage and drowning man classes contain the least number of images, which can lead to training anomalies. The distribution of images between datasets is shown in figure 9.

This distribution is due to the fact that for the available 6,000 images, the test data set will consist of three hundred images, which is more than enough to test the performance of the model.

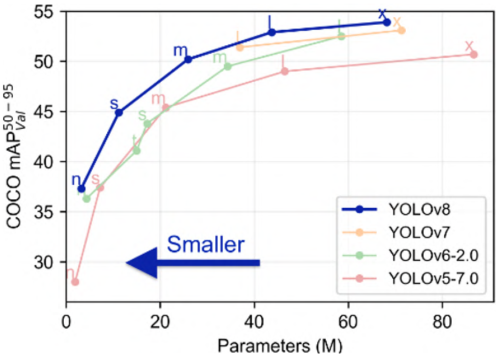
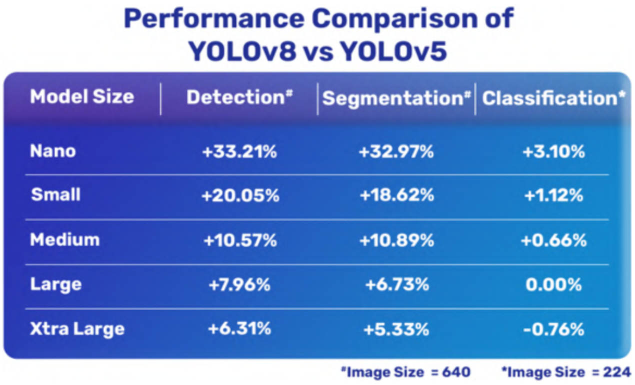


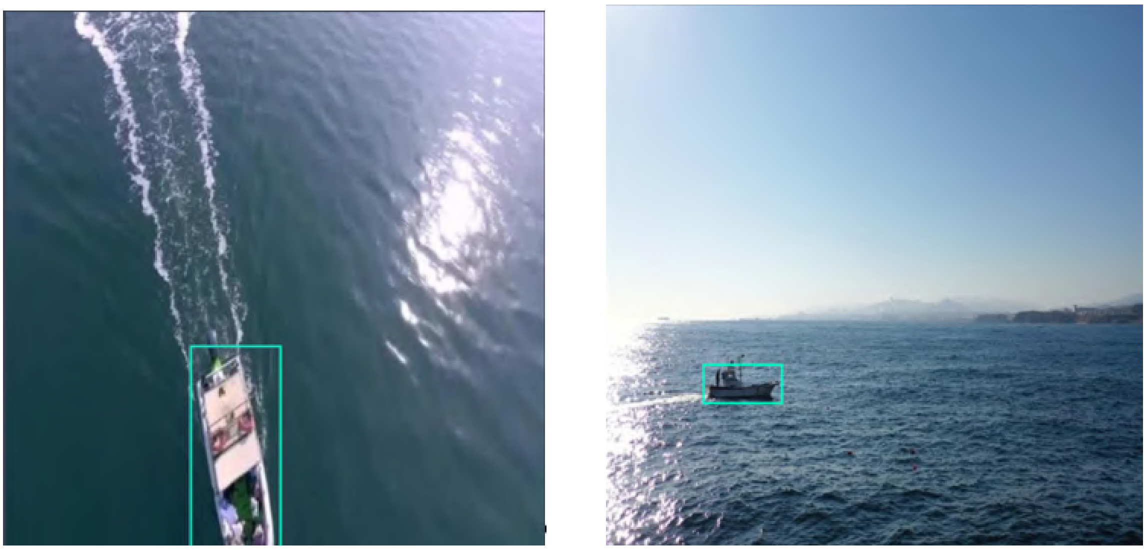
Figure 1: Average accuracy on the dataset common objects in context depending on the model used and the number of trained parameters.


Performance Comparison of YOLOv8 vs YOLOv5

Model Size	Detection*	Segmentation*	Classification*
Nano	+33.21%	+32.97%	+3.10%
Small	+20.05%	+18.62%	+1.12%
Medium	+10.57%	+10.89%	+0.66%
Large	+7.96%	+6.73%	0.00%
Xtra Large	+6.31%	+5.33%	-0.76%

*Image Size = 640 *Image Size = 224

Figure 2: Comparison of performance of YOLO (you only look once) v8 vs YOLOv5 models in detection, segmentation and classification tasks.



(a)

(b)

Figure 3: Examples of images of the "Boat" object class, (a) – top view and (b) – front view.

The selection of these classes for the dataset was due to the fact that it allows covering a large part of the objects of maritime navigation and interaction. The addition of human images to the dataset is also due to the fact that the deployable drone can be used as a rescue drone and add the functionality of calling rescuers or providing assistance: lifebuoys or vests can be attached to the drone.

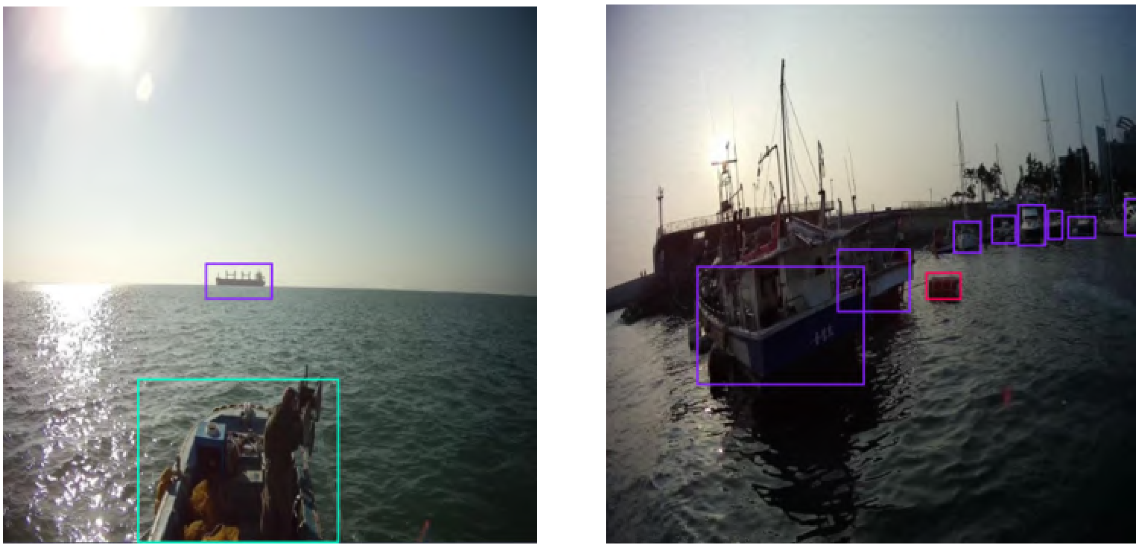
2.2. Primary testing

For the YOLOv5 model, the confusion matrix is shown in figure 10.

The history of the metrics of the training model is given in figure 11.

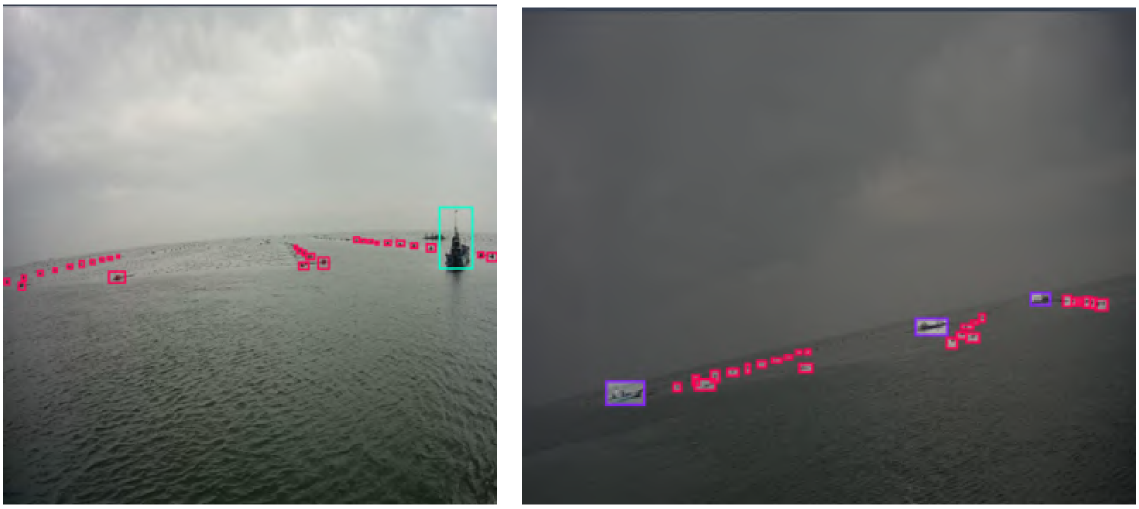
We can see that the model classifies swimmers and debris islands well, although this may be the result of insufficient images for these classes. The most successful class for recognition was "boat" with a probability of correct recognition of 52 percent, which is the expected result for this model. The classes "Buoy" and "Drowning man" are recognized worse and usually the model classifies them as background.

This may be due to both their similarity in the image and their small size and few special features. Ships are also poorly recognized, possibly due to the large number of images in the dataset, in which the ship is a tanker on the horizon and, accordingly, has small dimensions in the image.



(a) (b)

Figure 4: Examples of images of the “Ship” object class, (a) – single images of ships at a distance from each other and (b) – boats with a background and a certain number of the same and different types with and without accompanying objects.



(a) (b)

Figure 5: Examples of images of the object class “Buoy”, a), b) contain concentrated and distributed aggregates of buoys in a frontal view with background and extraneous objects. A distinctive feature of all markings is their small size.

Figure 12 shows the results of model verification.

The analysis shows that the model recognizes the object correctly, but gives a small percentage of confidence in its predictions. It is because of this fact that some objects remain unrecognizable.

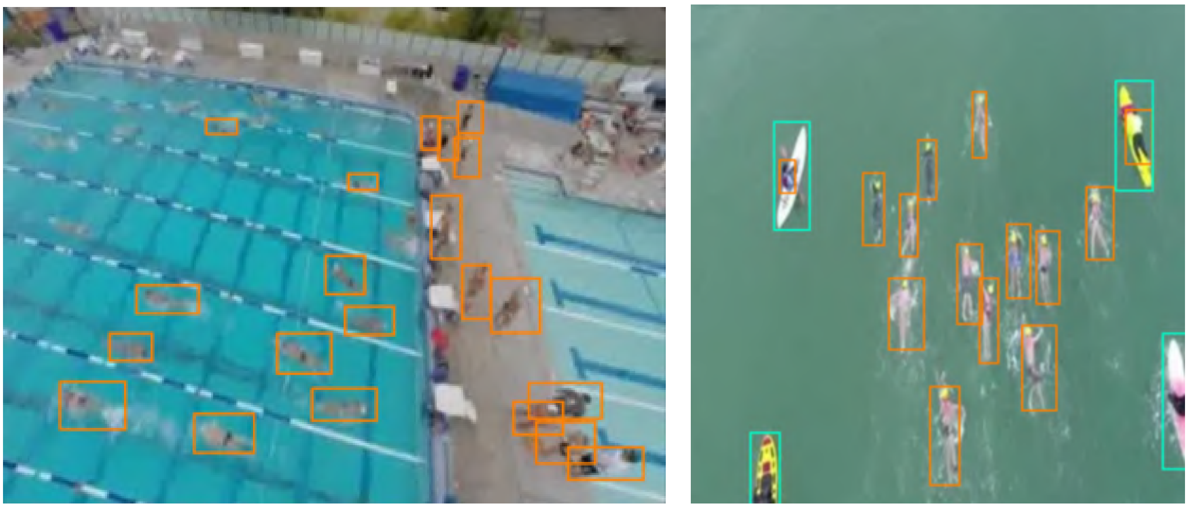
For the YOLOv8 model, figure 13 shows the confusion matrix.

The history of the metrics of the training model is given in figure 14.

Results of model verification is given in figure 15.

The analysis shows that the model recognizes the object correctly, but gives a small percentage of confidence in its predictions. It is because of this fact that some objects remain unrecognizable.

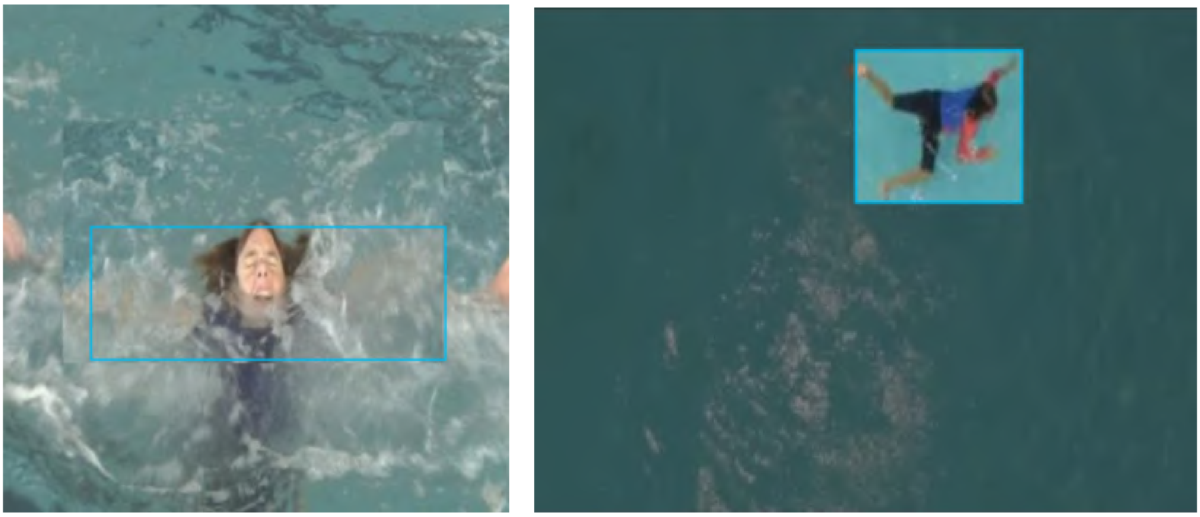
The obtained results give grounds for the conclusion that problems in training the model arise



(a)

(b)

Figure 6: Examples of images of the “Swimmer” object class, all images are shallow and a large cluster is noted for them, Figure 6 b) contains extraneous objects.



(a)

(b)

Figure 7: Examples of images of the object class “Drowning Man”, all images in Figure 7 are similar and almost indistinguishable from images of swimmers.

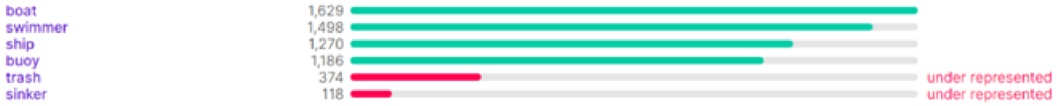


Figure 8: Distribution of images by classes.

precisely because of the dataset. Accordingly, for further development and obtaining a higher-quality model, an increase in the number and quality of marked images is required.

At the current stage, YOLO8 has better class recognition performance, although both models correctly locate objects, but have low confidence in the obtained results. This could be due to poor annotation, namely the similarity of the classes “Boat” and “Ship”, “Drowning man” and “Swimmer”, so I think it is

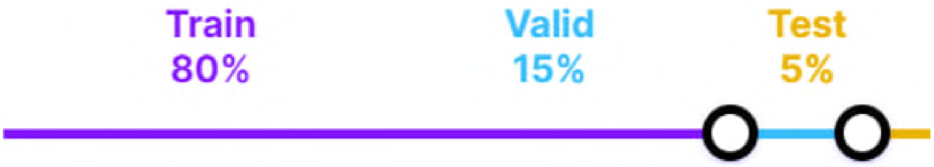


Figure 9: Distribution of images between datasets.

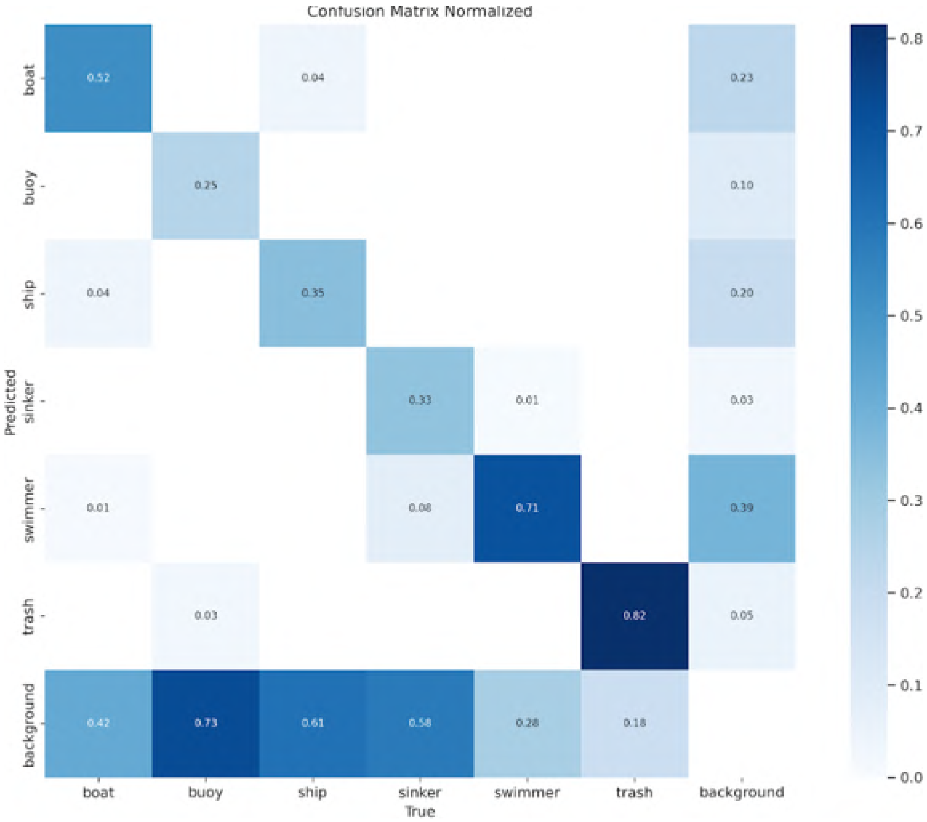


Figure 10: The confusion matrix.

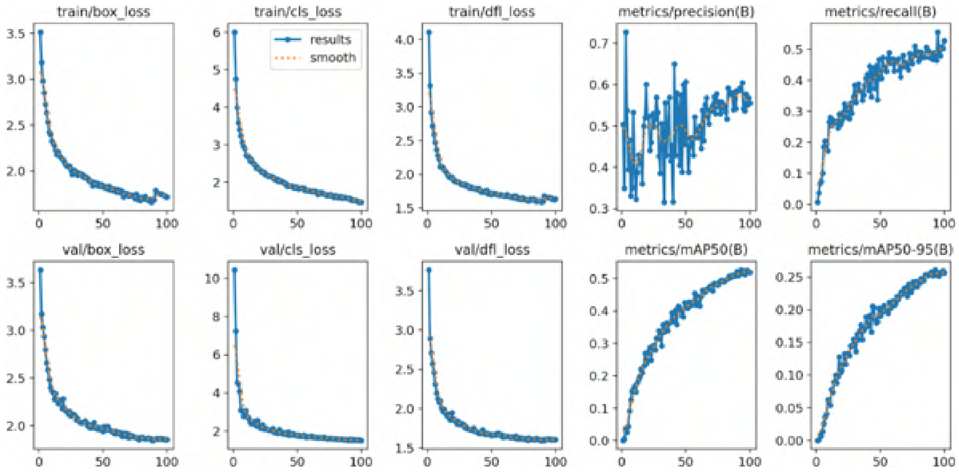
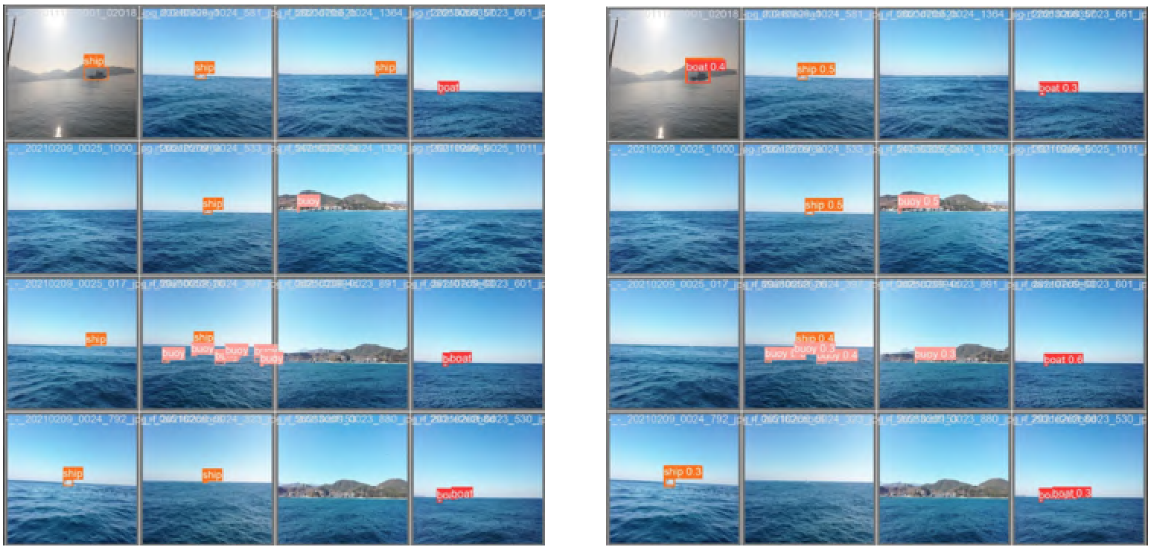


Figure 11: History of training model metrics.

necessary to add more images, re-evaluate the old ones and add data augmentation to diversify the training data.



(a) (b)

Figure 12: Results of model verification.

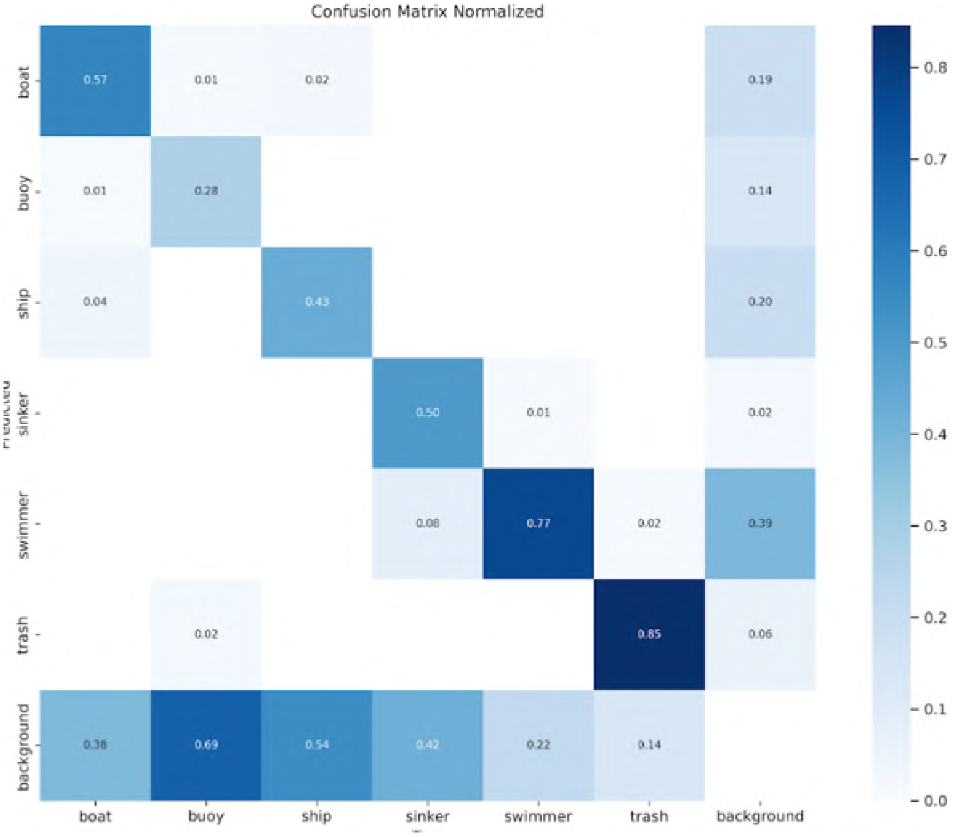


Figure 13: Confusion matrix for the YOLOv8 model.

3. Dataset editing

After marking another two thousand images, it was decided to move on to training a new model based on the YOLOv8 model. Images with the following types of augmentations were introduced into the dataset: cropping images for better behavior on images with small objects, generating a new image by

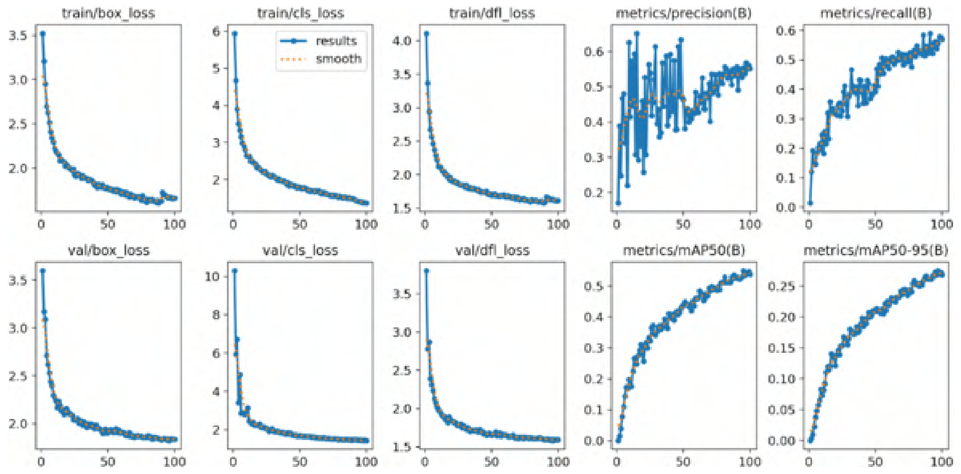
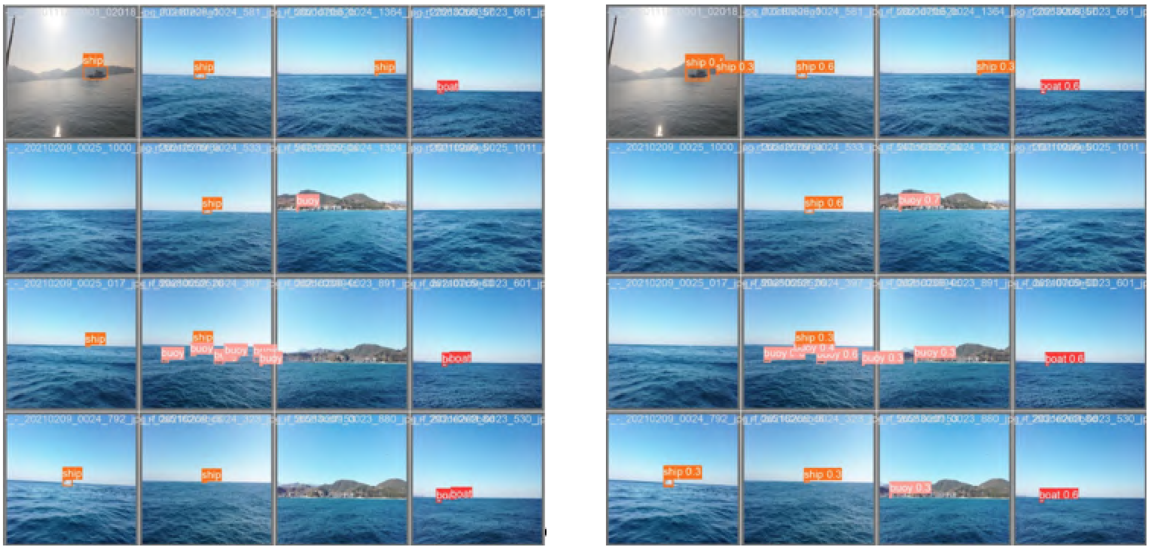


Figure 14: History of training model metrics.



(a)

(b)

Figure 15: Results of model verification. a) – from the dataset, b) – network prediction.

placing images in a mosaic, which allows the model to better process images with lower resolution and smaller objects, vertical mirroring in order to exclude the possibility of an uneven distribution of boats between the classes with the nose to the left and to the right.

The parameters of the resulting dataset are given in figure 16.

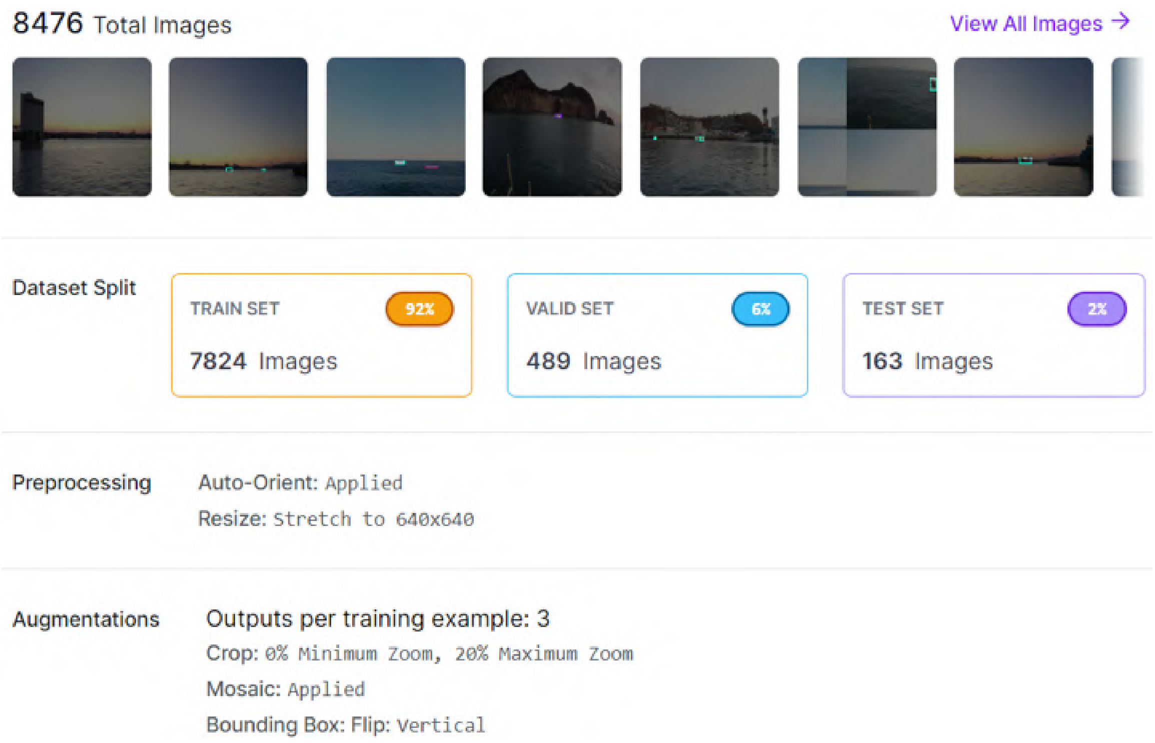
Examples of augmented images of objects are shown in figure 17.

Training was performed over one hundred epochs with a pre-trained model provided by the ultralytics API. In figure 18 shows the history of training metrics of the YOLOv8 model.

Figure 19 shows the confusion matrix for the YOLOv8 model.

The results of detection of the newly trained model are shown in figure 20.

We can see that the results differ to a certain extent, which confirms the correctness of the chosen direction of improving the dataset.



8476 Total Images

View All Images →

Dataset Split

TRAIN SET 92%

7824 Images

VALID SET 6%

489 Images

TEST SET 2%

163 Images

Preprocessing

Auto-Orient: Applied

Resize: Stretch to 640x640

Augmentations

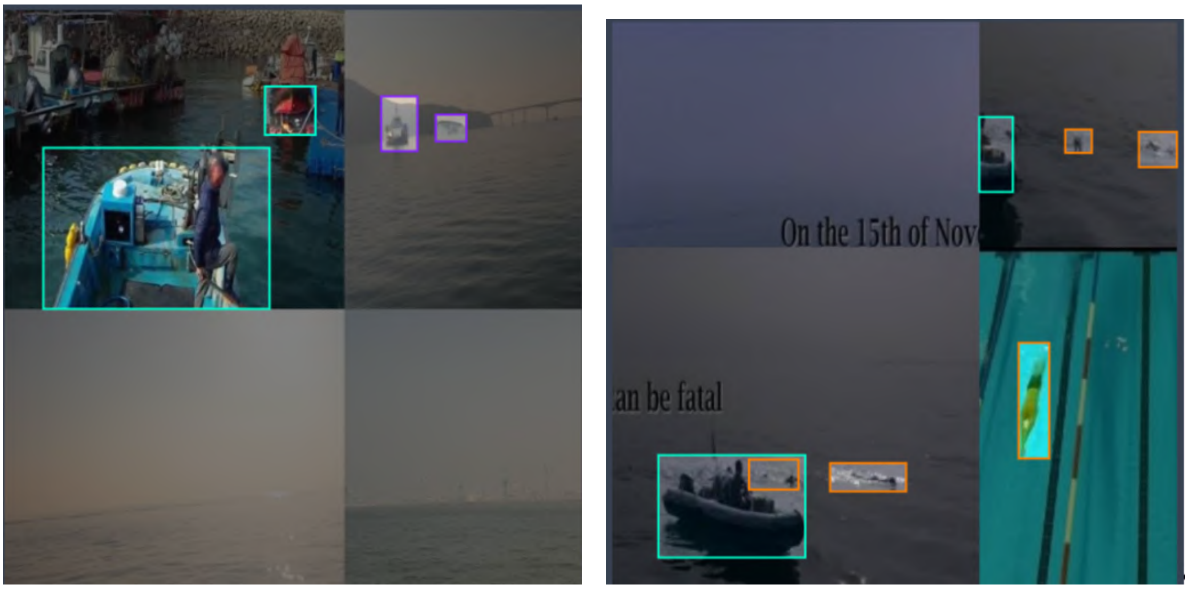
Outputs per training example: 3

Crop: 0% Minimum Zoom, 20% Maximum Zoom

Mosaic: Applied

Bounding Box: Flip: Vertical

Figure 16: Parameters of the resulting dataset.



(a)

(b)

Figure 17: A mosaic-augmented image of the object.

4. Testing

4.1. Testing on third-party images with classification enabled

In figure 21a shows the image after processing by the YOLOv8 model with an accuracy threshold of 0.4, which was able to detect the object “Swimmer” at the location of one of the many objects “Boat” with a probability of 0.56.

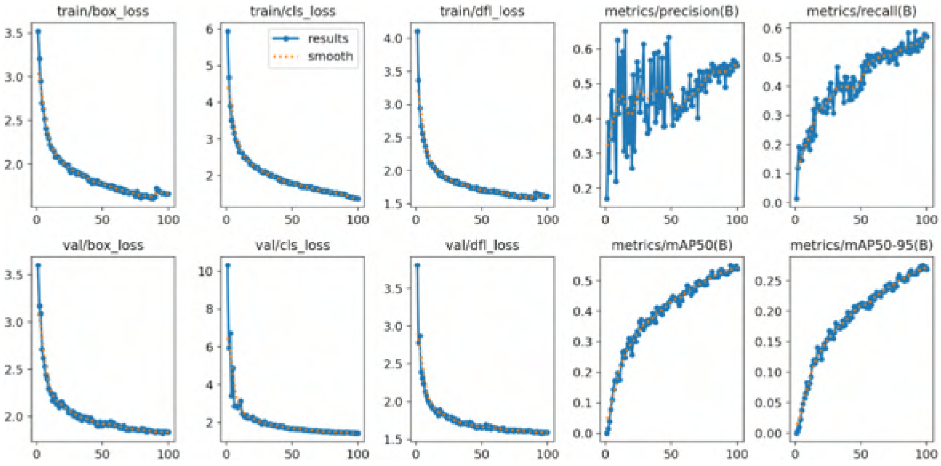
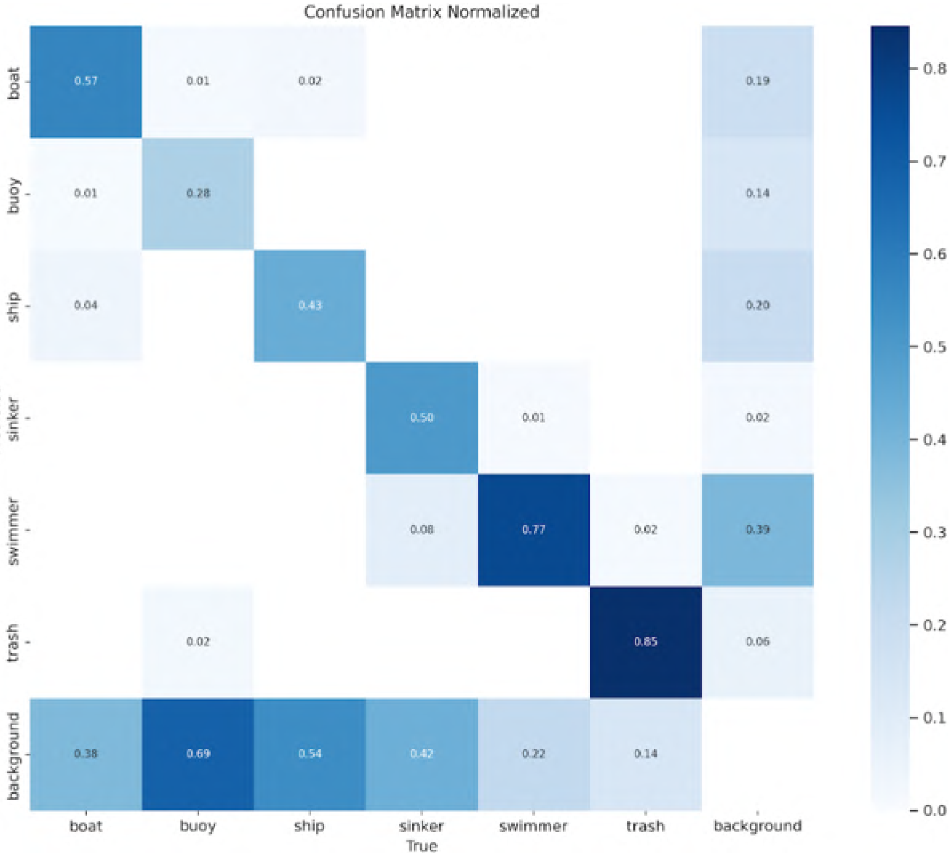


Figure 18: YOLOv8 model training metric history after dataset change.

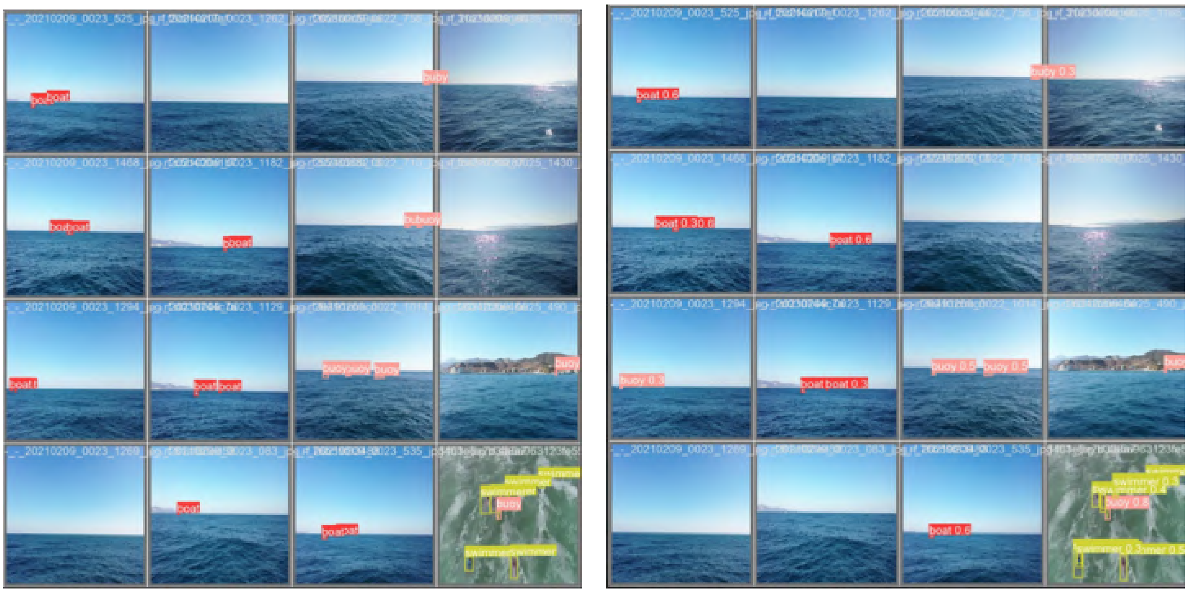


	boat	buoy	ship	sinker	swimmer	trash	background
boat	0.57	0.01	0.02				0.19
buoy	0.01	0.28					0.14
ship	0.04		0.43				0.20
sinker				0.50	0.01		0.02
swimmer				0.08	0.77	0.02	0.39
trash		0.02				0.85	0.06
background	0.38	0.69	0.54	0.42	0.22	0.14	

Figure 19: Confusion matrix for the YOLOv8 model.

The image shown in figure 21b has a high probability of recognizing the objects “Boat” and “Ship”, but assigns the recognized objects to the wrong class (misclassification).

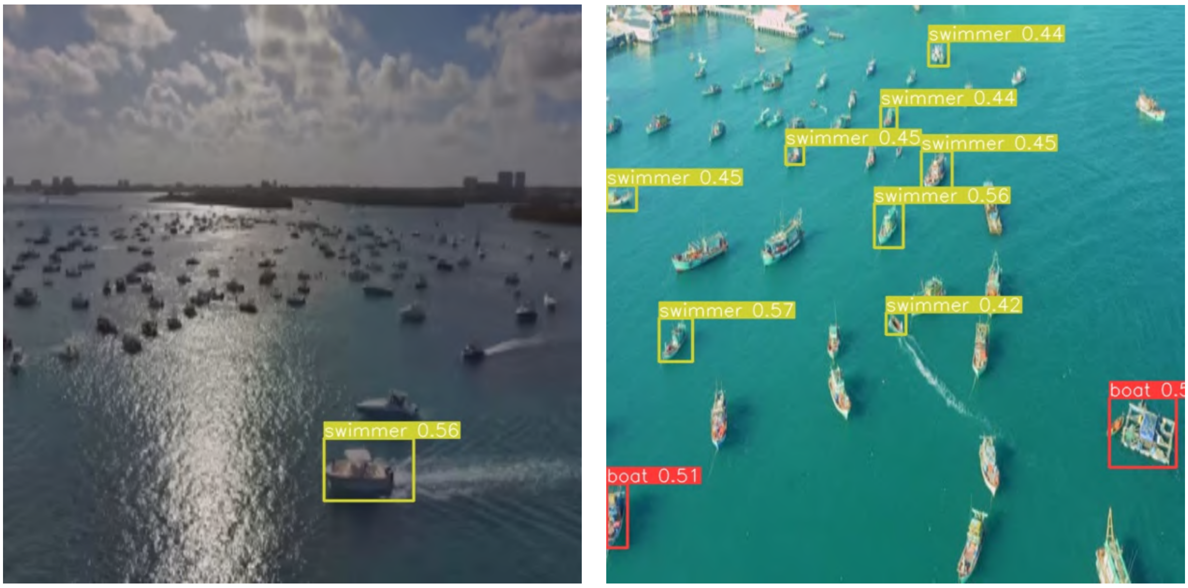
A fairly large volume of research was conducted, as a result of which a significant number of results were obtained and summarized, in particular. A single simple object in the frontal image is correctly recognized and classified. The small Buoy object in the background is completely ignored by the model. A small number of relatively large mountain-view objects were classified with high confidence and correctly. The swimmers closest to the camera were most likely detected, all other objects, including the “Boat”, were not detected. When testing the model, the Garbage object was not recognized, and the Boat object was completely ignored. The example of a swimming frame illustrates the correct recognition



(a)

(b)

Figure 20: Results of model verification: a) – marked, b) – predicted



(a)

(b)

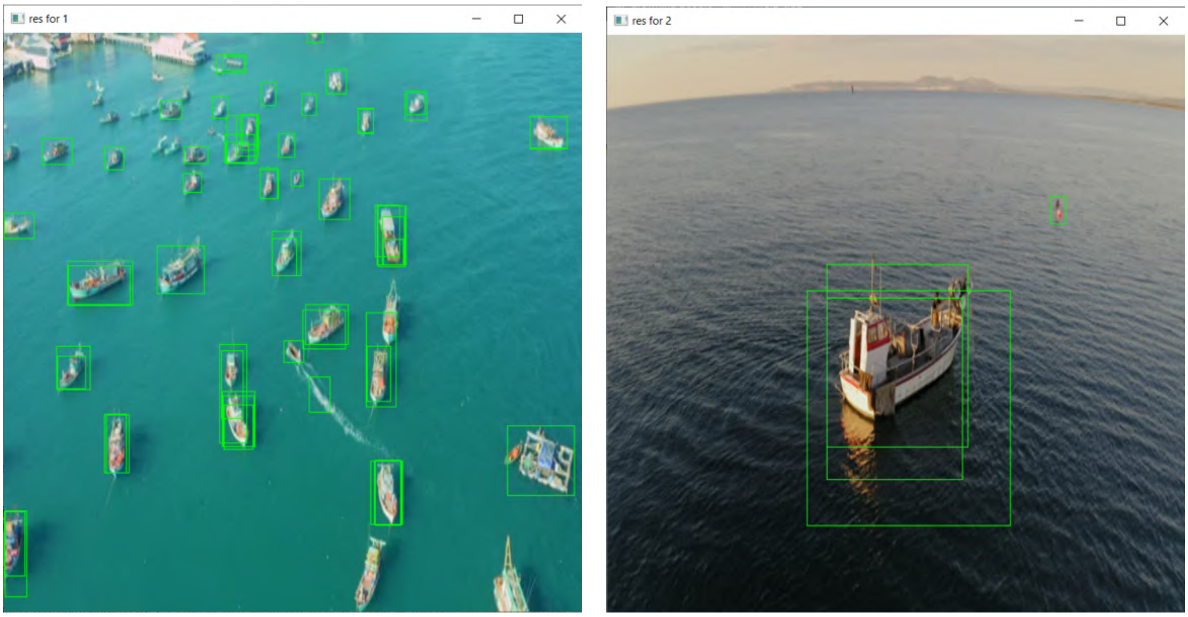
Figure 21: Image after processing with YOLOv8 model

of several Swimmer objects from a large number of available ones and the complete ignoring of the recognition of Boat objects. For the three Boat objects, the YOLOv8 image model sees two Swimmer objects instead of the expected Boat objects. Non-existent objects are not found, the class of existing objects is confused, possibly due to the small size of the latter.

4.2. Test results for reduced accuracy threshold with object detection classification function disabled, YOLOv8

For the following objects, the accuracy threshold was reduced to 0.1 and the classification function was disabled. Almost all recognized objects were correctly located. The model does not define objects where they do not exist.

The figure 22 shows examples of images with the accuracy threshold reduced to 0.1 and the classification function disabled.



(a)

(b)

Figure 22: Image with reduced accuracy threshold to 0.1 and disabled classification function.

For experimental objects, 100 percent of objects were marked regardless of scale. Marking occurred several times, which can be corrected by filtering the resulting bounding boxes.

In the presence of a large cluster of diverse and different objects, large-scale foreground objects were marked. The rest of the objects are consolidated into one large object.

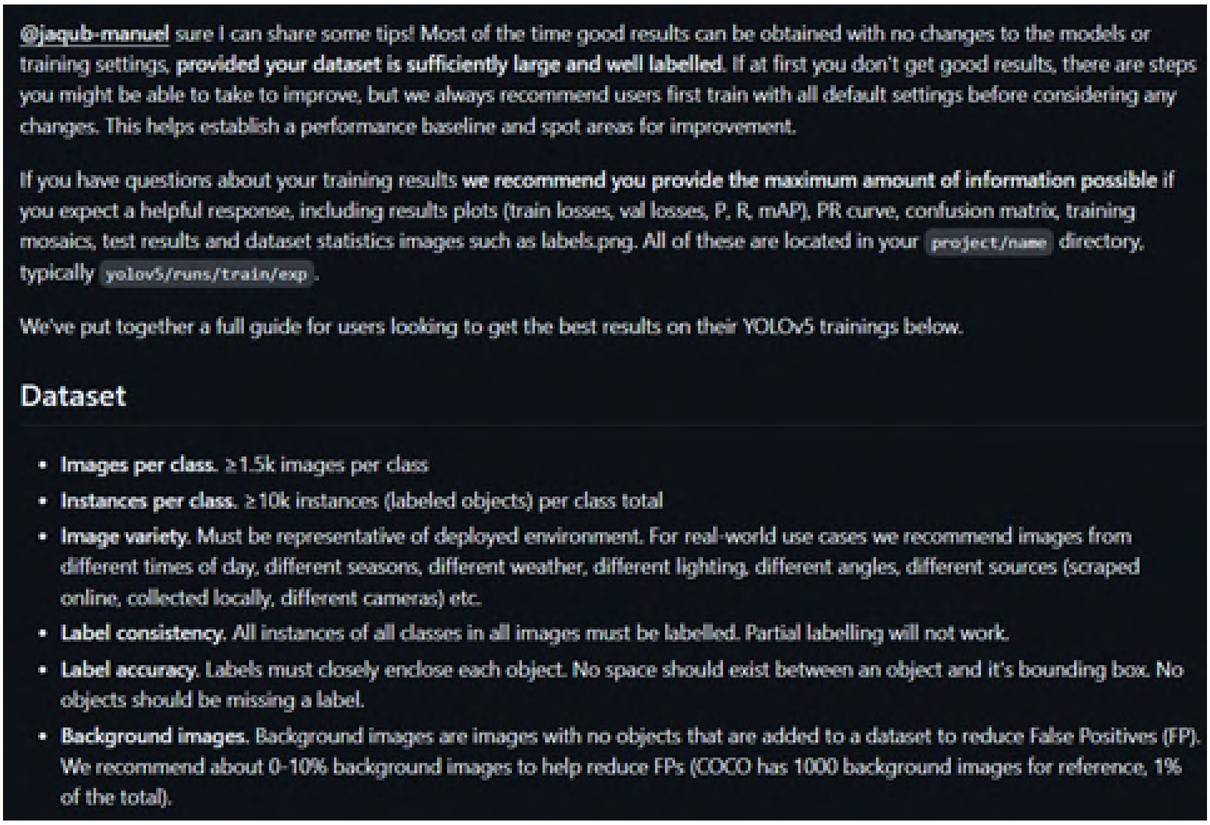
There were options where objects in the foreground only were recognized that were given a large scale, 100 percent of objects were marked regardless of scale, most of the various objects were marked regardless of scale, and non-existent objects were not marked. All obtained results are processed and summarized.

5. Conclusions

According to the obtained results, it can be seen that the system detects objects on the water surface, but the classification of these objects is not good. There are several reasons for this: errors in the labeling of the dataset and the small size of the dataset.

The comments shown in the figure 23 have been received from the API developers for working with artificial intelligence.

A possible scenario of using the built model is the general collection of information about the reservoir without regard to the classification output. In the process of such exploitation, it can be considered as expedient to collect a dataset that will correspond to the data from the drone (the data of the current dataset is data from surveillance cameras and video recordings from boats). In the future, form the dataset according to the developer's requirements, applying the necessary data augmentation steps.



@jaqub-manuel sure I can share some tips! Most of the time good results can be obtained with no changes to the models or training settings, **provided your dataset is sufficiently large and well labelled.** If at first you don't get good results, there are steps you might be able to take to improve, but we always recommend users first train with all default settings before considering any changes. This helps establish a performance baseline and spot areas for improvement.

If you have questions about your training results **we recommend you provide the maximum amount of information possible** if you expect a helpful response, including results plots (train losses, val losses, P, R, mAP), PR curve, confusion matrix, training mosaics, test results and dataset statistics images such as labels.png. All of these are located in your `project/name` directory, typically `yolov5/runs/train/exp`.

We've put together a full guide for users looking to get the best results on their YOLOv5 trainings below.

Dataset

- **Images per class.** ≥1.5k images per class
- **Instances per class.** ≥10k instances (labeled objects) per class total
- **Image variety.** Must be representative of deployed environment. For real-world use cases we recommend images from different times of day, different seasons, different weather, different lighting, different angles, different sources (scraped online, collected locally, different cameras) etc.
- **Label consistency.** All instances of all classes in all images must be labelled. Partial labelling will not work.
- **Label accuracy.** Labels must closely enclose each object. No space should exist between an object and it's bounding box. No objects should be missing a label.
- **Background images.** Background images are images with no objects that are added to a dataset to reduce False Positives (FP). We recommend about 0-10% background images to help reduce FPs (COCO has 1000 background images for reference, 1% of the total).

Figure 23: Developer comments.

6. Author contributions

The idea of writing the article belongs to all authors. Viktorija Smolij built and trained the model, Natan Smolij performed testing and analyzed the training results, Sergii Sayapin collected and marked up the dataset, performed the design of the materials.

Acknowledgments

When writing the article, special thanks should be expressed to the head of the department Oleksandr Rolik, Volodymyr Oliynyk, Yury Berdnyk and Mykola Shynkevych. Great gratitude for the explanation and organization of work, support and kindness.

References

[1] P. Schönfelder, F. Stebel, N. Andreou, M. König, Deep learning-based text detection and recognition on architectural floor plans, Automation in Construction 157 (2024) 105156. doi:10.1016/j.autcon.2023.105156.

[2] N. Giakoumoglou, E. M. Pechlivani, D. Tzovaras, Generate-Paste-Blend-Detect: Synthetic dataset for object detection in the agriculture domain, Smart Agricultural Technology 5 (2023) 100258. doi:10.1016/j.atech.2023.100258.

[3] M. M. Mintii, Exploring the landscape of STEM education and personnel training: a comprehensive systematic review, Educational Dimension 9 (2023) 149–172. doi:10.31812/ed.583.

[4] K. S. Tarisayi, Strategic leadership for responsible artificial intelligence adoption in higher education, CTE Workshop Proceedings 11 (2024) 4–14. doi:10.55056/cte.616.

[5] M. Ashourpour, G. Azizpour, K. Johansen, Real-Time Defect and Object Detection in Assembly Line: A Case for In-Line Quality Inspection, in: F. J. G. Silva, A. B. Pereira, R. D. S. G. Campilho (Eds.), Flexible Automation and Intelligent Manufacturing: Establishing Bridges for More Sustainable Manufacturing Systems, Springer Nature Switzerland, Cham, 2024, pp. 99–106. doi:10.1007/978-3-031-38241-3_12.

[6] P. Azevedo, V. Santos, Comparative analysis of multiple YOLO-based target detectors and trackers for ADAS in edge devices, Robotics and Autonomous Systems 171 (2024) 104558. doi:10.1016/j.robot.2023.104558.

[7] M. Sanjai Siddharthan, S. Aravind, S. Sountharrajan, Real-Time Road Hazard Classification Using Object Detection with Deep Learning, in: P. P. Joby, M. S. Alencar, P. Falkowski-Gilski (Eds.), IoT Based Control Networks and Intelligent Systems, Springer Nature, Singapore, 2024, pp. 479–492. doi:10.1007/978-981-99-6586-1_33.

[8] N. V. Morze, O. V. Strutynska, Advancing educational robotics: competence development for pre-service computer science teachers, CTE Workshop Proceedings 10 (2023) 107–123. doi:10.55056/cte.549.

[9] Z. H. Wei, Y. J. Zhang, X. J. Wang, J. T. Zhou, F. Q. Dou, Y. H. Xia, A YOLOv8-based approach for steel plate surface defect detection, Metalurgija 63 (2024) 28–30.

[10] F. Wu, Y. Zhang, L. Wang, Q. Hu, S. Fan, W. Cai, A Deep Learning-Based Lightweight Model for the Detection of Marine Fishes, Journal of Marine Science and Engineering 11 (2023) 2156. doi:10.3390/jmse11112156.

[11] G. Zhang, Y. Tang, H. Tang, W. Li, L. Wang, A global lightweight deep learning model for express package detection, Journal of Intelligent and Fuzzy Systems 45 (2023) 12013–12025. doi:10.3233/JIFS-232874.

[12] J. Wang, H. Dai, T. Chen, H. Liu, X. Zhang, Q. Zhong, R. Lu, Toward surface defect detection in electronics manufacturing by an accurate and lightweight YOLO-style object detector, Scientific Reports 13 (2023) 7062. doi:10.1038/s41598-023-33804-w.

[13] A. Li, Z. Zhang, S. Sun, M. Feng, C. Wu, MultiNet-GS: Structured Road Perception Model Based on Multi-Task Convolutional Neural Network, Electronics 12 (2023) 3994. doi:10.3390/electronics12193994.

[14] L. Han, C. Ma, Y. Liu, J. Jia, J. Sun, SC-YOLOv8: A Security Check Model for the Inspection of Prohibited Items in X-ray Images, Electronics 12 (2023) 4208. doi:10.3390/electronics12204208.

[15] A. R. Petrosian, R. V. Petrosyan, I. A. Pilkevych, M. S. Graf, Efficient model of PID controller of unmanned aerial vehicle, Journal of Edge Computing 2 (2023) 104–124. doi:10.55056/jec.593.

[16] J. Mao, L. Wang, N. Wang, Y. Hu, W. Sheng, A novel method of human identification based on dental impression image, Pattern Recognition 144 (2023) 109864. doi:10.1016/j.patcog.2023.109864.

[17] E. Kara, G. Zhang, J. J. Williams, G. Ferrandez-Quinto, L. J. Rhoden, M. Kim, J. N. Kutz, A. Rahman, Deep learning based object tracking in walking droplet and granular intruder experiments, Journal of Real-Time Image Processing 20 (2023) 86. doi:10.1007/s11554-023-01341-4.

[18] S. Zhou, M. Zhong, X. Chai, N. Zhang, Y. Zhang, Q. Sun, T. Sun, Framework of rod-like crops sorting based on multi-object oriented detection and analysis, Computers and Electronics in Agriculture 216 (2024) 108516. doi:10.1016/j.compag.2023.108516.

[19] P. Shan, R. Yang, H. Xiao, L. Zhang, Y. Liu, Q. Fu, Y. Zhao, UAVPNet: A balanced and enhanced UAV object detection and pose recognition network, Measurement: Journal of the International Measurement Confederation 222 (2023) 113654. doi:10.1016/j.measurement.2023.113654.

[20] S. O. Semerikov, T. A. Vakaliuk, I. S. Mintii, V. A. Hamaniuk, V. N. Soloviev, O. V. Bondarenko, P. P. Nechypurenko, S. V. Shokaliuk, N. V. Moiseienko, V. R. Ruban, Development of the computer vision system based on machine learning for educational purposes, Educational Dimension 5 (2021) 8–60. doi:10.31812/educdim.4717.

[21] F. M. Talaat, H. ZainEldin, An improved fire detection approach based on YOLO-v8 for smart cities, Neural Computing and Applications 35 (2023) 20939–20954. doi:10.1007/s00521-023-08809-1.

[22] S. Liu, Q. Fan, C. Zhao, S. Li, RTAD: A Real-Time Animal Object Detection Model Based on a Large Selective Kernel and Channel Pruning, Information 14 (2023) 535. doi:10.3390/info14100535.

[23] V. Smolij, About features of management preproduction of electronic vehicles, Problems of Modeling and Design Automatization 11 (2019). doi:10.31474/2074-7888-2019-1-33-42.

[24] Y. Su, W. Tan, Y. Dong, W. Xu, P. Huang, J. Zhang, D. Zhang, Enhancing concealed object detection in Active Millimeter Wave Images using wavelet transform, Signal Processing 216 (2024) 109303. doi:10.1016/j.sigpro.2023.109303.

[25] C. Liu, K. Wang, Q. Li, F. Zhao, K. Zhao, H. Ma, Powerful-IoU: More straightforward and faster bounding box regression loss with a nonmonotonic focusing mechanism, Neural Networks 170 (2024) 276–284. doi:10.1016/j.neunet.2023.11.041.

[26] W. Xu, C. Liu, G. Wang, Y. Zhao, J. Yu, A. Muhammad, D. Li, Behavioral response of fish under ammonia nitrogen stress based on machine vision, Engineering Applications of Artificial Intelligence 128 (2024) 107442. doi:10.1016/j.engappai.2023.107442.

[27] G. Dimauro, N. Barbaro, M. G. Camporeale, V. Fiore, M. Gelardi, M. Scalera, DeepCilia: Automated, deep-learning based engine for precise ciliary beat frequency estimation, Biomedical Signal Processing and Control 90 (2024) 105808. doi:10.1016/j.bspc.2023.105808.

[28] X. Zhao, Y. Song, Improved Ship Detection with YOLOv8 Enhanced with MobileViT and GSConv, Electronics 12 (2023) 4666. doi:10.3390/electronics12224666.

An analysis of approach to the features of satellites classification determining based on modeling of linguistic variables and membership functions

Ihor A. Pilkevych[1], Iryna A. Bespalko[1], Leonid M. Naumchak[1] and Dmytro V. Pekariev[2]

[1] *Korolyov Zhytomyr Military Institute, 22 Myru Ave., Zhytomyr, 10004, Ukraine*

[2] *Section of applied problems of the Presidium of the National Academy of Sciences of Ukraine, 54 Volodymyrska Str., Kyiv, 02000, Ukraine*

Abstract

The modern approaches to the classification of satellites was analyzed, the relevance of the use of the fuzzy logic apparatus and the main stages of solving the given problem using the theory of fuzzy sets was determined. The features of the classification of satellites, which can be obtained both from the analysis of a priori and a posteriori information about satellites, and can be numerical, categorical or linguistic, was determined. The need to define linguistic variables and their linguistic terms for those features of the classification of satellites that can be presented in a linguistic form was substantiated. The choice of the method of constructing the membership function of a fuzzy set of defined features of the satellites classification, which can be presented in a linguistic form, was justified. Further steps to solve the problem of satellites classification based on fuzzy logic was outlined: building a system of fuzzy rules for satellites identification and creating a fuzzy knowledge base for their classification.

Keywords

classification of satellites, features of classification, fuzzy set, linguistic variables, membership function,

1. Introduction

The composition of the space systems of the world's leading states that carry out space activities is actively changing today. The number of satellites is increasing, their functional capabilities are improving due to the development of the material, technical and scientific base.

Considering the martial law introduced in Ukraine from February 24, 2022, space support and, in particular, space situational awareness (space situation analysis) is an urgent need in the process of planning the activities of national security and defense entities, which requires a clear classification of satellites, which is determined by their purpose.

A clear understanding of the purpose of satellites allows you to take into account the peculiarities of their functioning and influence on various spheres of activity of state authorities, especially to ensure the national security and defense of Ukraine [1]. It is necessary to classify satellites, which is an important task for carrying out space activities (for example, planning observations, protection against possible observations from space, etc.).

The development of technologies and the appearance of new satellites may require the expansion of existing classification features, that is, such as the satellites classes defined for a certain period of time which are not static. Under such conditions, classification features obtained from both a priori and a posteriori information about satellites can be numerical, categorical or linguistic.

Taking into account heterogeneous features requires the use of appropriate mathematical apparatus, which will allow them to be formalized for the further classification of satellites, which is an current scientific task.

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine

igor.pilkevich@meta.ua (I. A. Pilkevych); biran198311@gmail.com (I. A. Bespalko); naumchak.leonid@gmail.com (L. M. Naumchak); dmvlpek@ukr.net (D. V. Pekariev)
https://ieeexplore.ieee.org/author/37089181628 (I. A. Pilkevych)
000-0001-5064-3272 (I. A. Pilkevych); 0000-0002-7197-7324 (I. A. Bespalko); 0000-0002-7311-6659 (L. M. Naumchak); 0000-0002-5056-4993 (D. V. Pekariev)

2. Related works

Many scientific works are devoted to the issue of object classification. Different approaches have been proposed to solve the classification task, for example, using of the backpropagation algorithm of artificial neural networks for the classification of GPS satellites and the calculation of geometric accuracy coefficients of their positioning [2], deep and multi-core learning based on recurrent and convolutional neural networks [3, 4] for synchronous identification of the shape and position of satellites in geostationary orbit [5], etc.

But most of the attention is given to the classification of satellites from the point of view of their further application or the use of data that can be obtained from satellites.

Thus, the classification of GPS satellites using improved learning algorithms is considered to solve the problem of calculating the geometric accuracy coefficients of GPS satellites positioning [2]. The unified classification of satellites based on mass and size is one of the tools for determining the size of launch vehicles and the cost of launching satellites into orbit [6]. Classification of satellites in geostationary orbit with deep and multi-core learning is one of the approaches to ensure the safety of objects in geostationary orbit [5].

In Ukrainian works, options for the satellites classification are considered using the example of species observation satellites based on the analysis of their features and the systematization of information about space systems, a generalized classification of satellites is proposed [7, 8, 9]. In other publications, attention is paid to the problems of choosing satellites for the use of their target information [10, 11, 12].

Thus, in modern scientific works, the results of research on the classification of satellites by individual features are reflected, and the specified task by a set of features is almost not considered.

In the case when there is no clear boundary separating the classes (for example, heterogeneous features belong to several classes), the approach using fuzzy logic [13, 14, 15] will allow classifying satellites by a set of heterogeneous features with a certain probability of truth [16].

The *purpose* of the article is to determine the linguistic variables and the membership function of a set of features for the further satellites classification using the theory of fuzzy sets.

3. Method

In the modern conditions of using information about the state and changes of the space situation, there is an urgent need for reliable and complete information about the purpose of satellites, which is complicated by certain limitations in the use of measuring tools, etc. [1, 17].

In order to increase the accuracy of determining the purpose of satellites, the reliability of their classification, it is proposed to use the mathematical apparatus of the theory of fuzzy sets to classify satellites based on a priori and a posteriori information that can be obtained from open sources.

The initial stage in the task of satellites classification using fuzzy set theory is the determination of the features of satellites that will be used for classification. These features can be numerical, categorical or linguistic.

It is possible to classify satellites according to the information that precedes their launch and the information that is available for analysis after the launch. Thus, it is possible to distinguish a priori (pre-launch) and a posteriori (post-launch) features of classification, which, in turn, can be direct and indirect [18].

The initial information before launch for classification is the satellite launch plan. Information from the satellite launch plan can be interpreted accordingly to table 1.

After launch, the satellite classification is refined based on the use of a posteriori information and its orbital parameters obtained from official sources or from measuring devices.

Taking into account that all features are different, it is possible to obtain a generalized conclusion and make a decision regarding the belonging of satellite to a certain class with a certain degree of truth using a mathematical apparatus of fuzzy derivation.

The problem of data classification can be solved by the fuzzy inference system, which is based on the

Table 1
Information from the satellite launch plan.

Category of information	Type of a priori feature
Declared purpose of the satellite	direct, categorical
The launch site (cosmodrome)	indirect, linguistic
The type of launch vehicle that will be used to launch the satellite	indirect, linguistic
Name of satellite	direct, linguistic
The customer of the satellite	indirect, linguistic
The developer of the satellite	indirect, linguistic
Configuration of the satellite	direct, linguistic
Launch mass of the satellite	indirect, numerical
Estimated (warranty) period of operation of the satellite	indirect, numerical
Type of orbit	indirect, linguistic
Inertial longitude of the ascending node of the orbit	indirect, numerical

algorithm of obtaining fuzzy conclusions based on fuzzy premises using concepts of fuzzy logic [16]. The process of fuzzy derivation combines the main concepts of fuzzy set theory: membership functions, linguistic variables, fuzzy logical operations, methods of fuzzy implication, and fuzzy composition [18].

The general scheme of the fuzzy inference system is presented in figure 1.

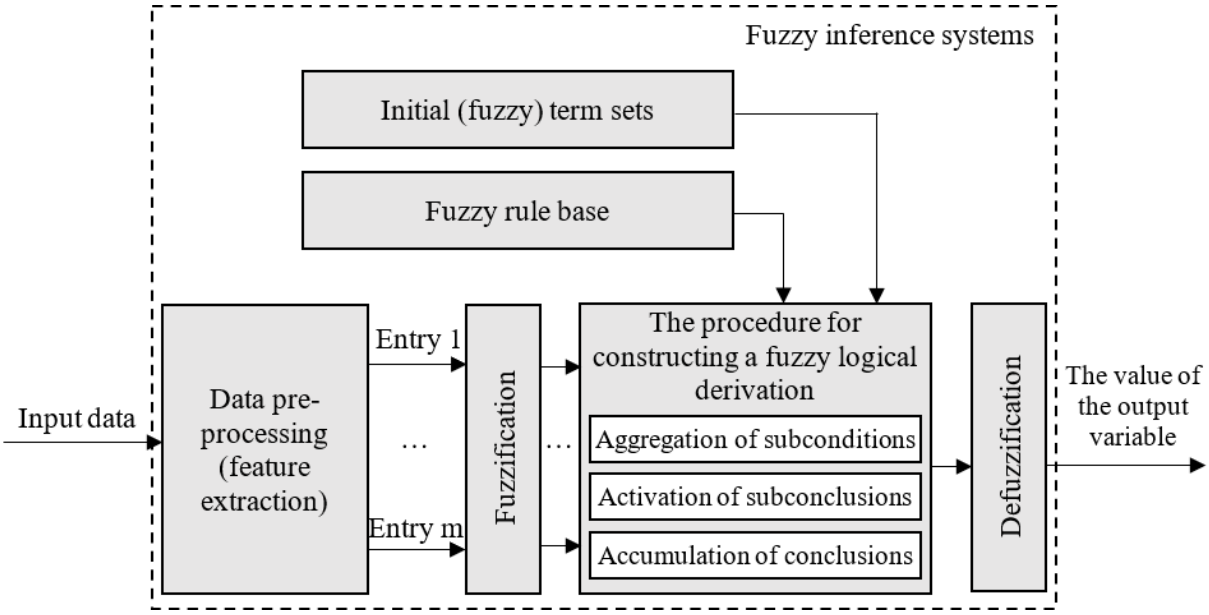

Fuzzy inference systems
Initial (fuzzy) term sets
Fuzzy rule base
Input data
Data pre-processing (feature extraction)
Entry 1
…
Entry m
Fuzzification
…
The procedure for constructing a fuzzy logical derivation
Aggregation of subconditions
Activation of subconclusions
Accumulation of conclusions
Defuzzification
The value of the output variable

Figure 1: The general scheme of the fuzzy inference system.

Fuzzy inference systems are defeaturesed to transform the values of input variables into output variables based on the use of fuzzy rules. For this, fuzzy inference systems should contain a base of fuzzy rules and initial term sets [18].

The main stages of fuzzy derivation (figure 1) are [18]:

- fuzzification of input variables;
- aggregation of preconditions in fuzzy rules;
- activation or composition of subconclusions in fuzzy rules;
- accumulation of conclusions of fuzzy rules.

In general, the classification of objects based on fuzzy logic is a complex process and requires a large amount of input data, but the main advantage of applying the proposed approach is the ability to use information that may be fuzzy, but still useful for decision-making.

Consider the first stage of the process of fuzzy derivation – fuzzification of input variables – establishing correspondence between the specific (usually numerical) value of a separate input variable of the system of fuzzy derivation and the value of the membership function of the corresponding term of the input linguistic variable. After that, specific values of membership functions for each of the linguistic terms used in the prerequisites of the fuzzy inference system rule base must be determined for all input variables [18].

Formally, the fuzzification procedure is performed as follows. At the beginning of fuzzification, the specific values of all input variables of the fuzzy inference system are determined, that is, the set of values $A = a^1, a^2, ..., a^m$.

In the general case, each $a_i \in E_i$, where E_i is the universe of the linguistic variable β_i.

Next, we consider each of the subconditions of the form $\beta_i \in T$ of the fuzzy derivation system rules, where T is some term with the corresponding membership function $\mu(x)$, which can be analytically specified, for example, in the following form:

$$\mu(x, a, b) = \left\{ \begin{array}{c} 1, x \leqslant a \\ \frac{b-x}{b-a}, a < x < b \\ 0, b \leqslant x \end{array} \right\}, \tag{1}$$

or

$$\mu(x, a, b) = \left\{ \begin{array}{c} 1, x \leqslant a \\ \frac{x-a}{b-a}, a < x < b \\ 0, b \leqslant x \end{array} \right\}. \tag{2}$$

At the same time, the value a_i is used as an argument of $\mu(x)$ and the quantitative value is found, which is the result of fuzzification of the subcondition.

The specified approach can be used to solve the task of satellites classification taking into account the majority of disparate features.

4. Experimental results

Suppose that the launch of the ViaSat 3.2 satellite (ViaSat 3 EMEA) is planned for 2024, which is about 6.4 tons, using an Atlas-5 launch vehicle in an orbit with an altitude of about 35,790 km [19].

For example, consider the features "Type of launch vehicle" and "Type of orbit", which are indirect linguistic a priori features for further classification of the satellites.

Correspondence between the type of launch vehicle and its payload is shown in table 2 [19, 20, 21].

Table 2

Correspondence between the type of launch vehicle and its payload.

Type of launch vehicle	Payload, tons
Small	up 2
Medium	2-20
Heavy	20-50
Overweight	> 50

Correspondence between the type of orbit and its altitude is shown in table 3 [19, 20, 21].

Let's define the linguistic variable "Type of launch vehicle" as β_1. Then "Small, Medium, Heavy, Overweight" will be the set of terms T_1 of this linguistic variable β_1:

$$T_1 = \{Small, Medium, Heavy, Overweight\}. \tag{3}$$

The set of all ranges of values of the variable β_1:

$$E_1 = [0, > 50]. \tag{4}$$

Table 3
Correspondence between the type of orbit and its altitude.

Type of launch vehicle	Payload, tons
Low	160-2000
Medium	2000-35786
High	> 35786

Let's define the linguistic variable "Type of orbit" as β_2. Then "Low, Medium, High" will be the set of terms T_2 of this linguistic variable β_2:

$$T_2 = \{Low, Medium, High\}\,. \tag{5}$$

The set of all ranges of values of the variable β_2:

$$E_2 = [160, > 35786]\,. \tag{6}$$

Consider the process of fuzzification of four fuzzy statements for the input linguistic variable β_1 – "Type of launch vehicle": "Type of launch vehicle small", "Type of launch vehicle medium", "Type of launch vehicle heavy", "Type of launch vehicle overweight". The fuzzification of the first fuzzy statement gives the value "0", which is obtained by substituting the value $x_1 = 6.4$ into of the argument of the membership function. The fuzzification of the second fuzzy statement gives the value "0.24", which is obtained by substituting the value $x_1 = 6.4$ into the argument of the function accessories. The fuzzification of the third and fourth fuzzy statement gives the value "0", which is obtained by substituting the value $x_1 = 6.4$ into the argument of the function accessories. The result of fuzzification for the input linguistic variable β_1 on figure 2.

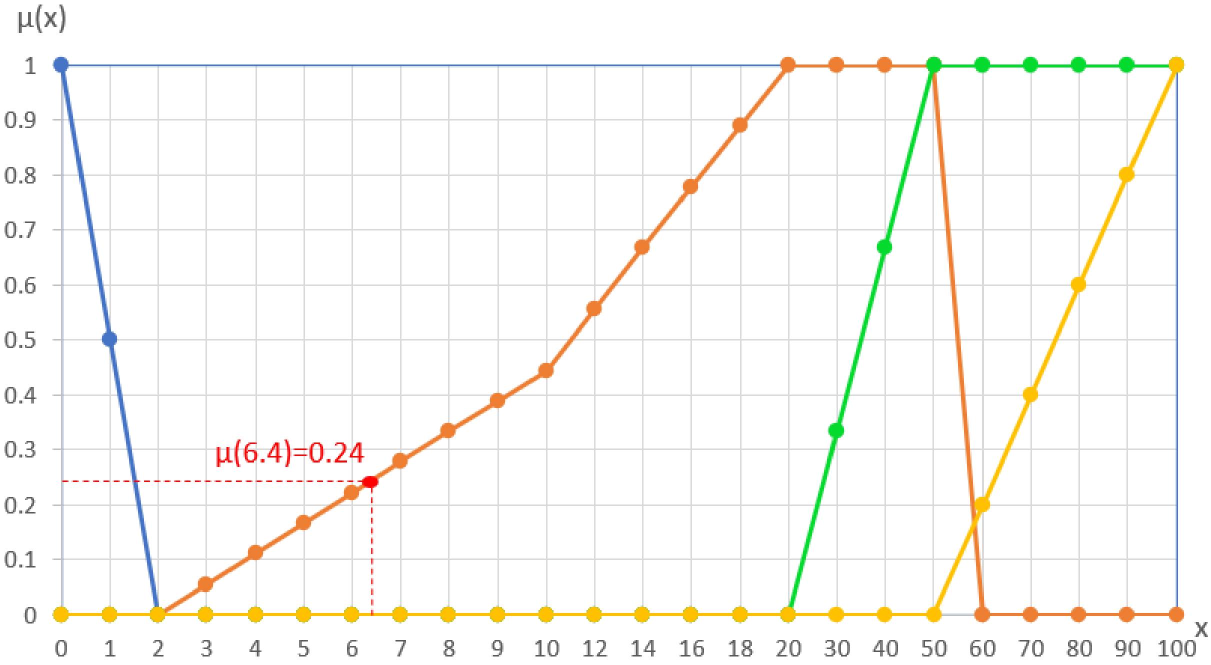


Figure 2: The result of fuzzification for the input linguistic variable β_1.

Consider the fuzzification process of three fuzzy statements for the input linguistic variable β_2 – "Type of orbit": "Orbit type is low", "Orbit type is medium", "Orbit type is high". The fuzzification of the first and second fuzzy statements gives the value "0", which is obtained by substituting the value $x_2 = 35790$ to the argument of the membership function for linguistic variable "Orbit type is low" and "Orbit type is medium". The fuzzification of the third fuzzy statement gives the value "0.84", which is obtained by substituting the value $x_2 = 35790$ to the argument of the membership function for linguistic variable “Orbit type is high”. The result of fuzzification for the input linguistic variable β_1 on figure 3.

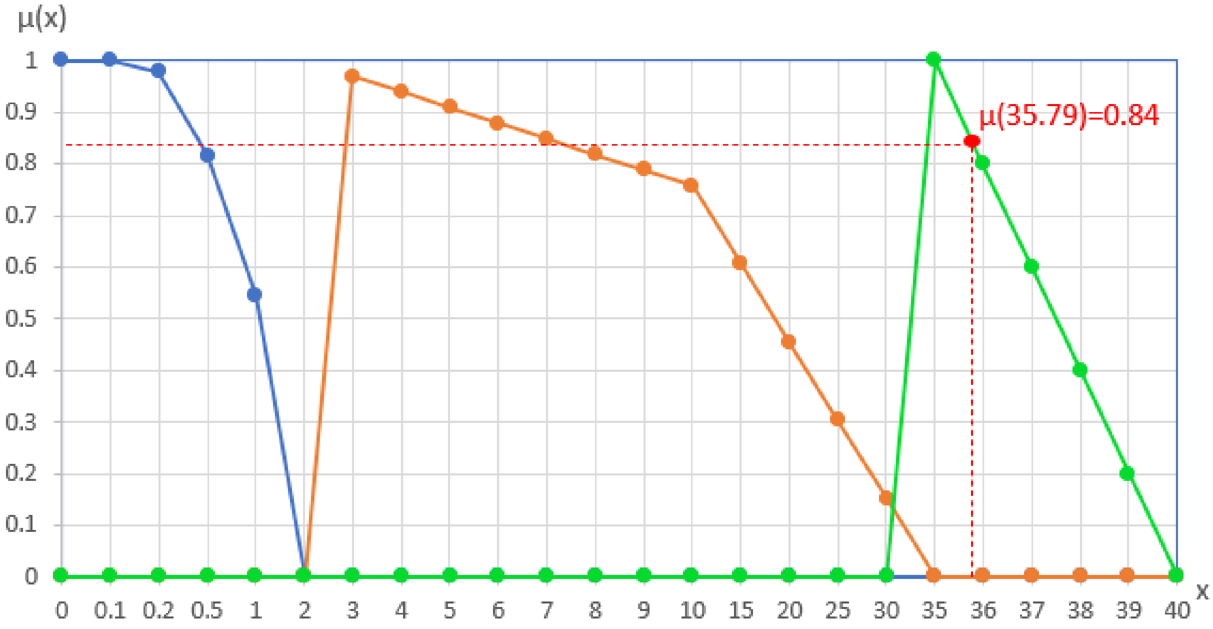


Figure 3: The result of fuzzification for the input linguistic variable β_2.

With the known values of the variables “Payload weight” = 6.4 tons and “Orbital height” = 35790 km, a preliminary conclusion can be made about the type of launch vehicle that can be used during the launch of the satellites and the likely type of orbit to which it will be possible the satellites will be launched.

5. Conclusions and further research

Thus, using the theory of fuzzy sets, the linguistic variables of some features of the satellites classification were formalized and an example of the calculation of their membership functions was given. The following steps in the classification process are:

1) finding the degrees of truth of the simplest statements based on the given values of the input parameters;
2) calculation of the truth of the prerequisites of the rules;
3) determination of membership functions of each of the conclusions for the general linguistic variable;
4) unification of membership functions through the construction of their maximum;
5) obtaining a specific value of the output variable.

The proposed approach can be used to solve the problem of complex classification of satellites, taking into account the majority of heterogeneous features.

6. Contributions by authors

The author’s contribution to the article is distributed as follows:

- Conceptualisation of reseach, formulation of the research idea, Dmytro V. Pekariev and Iryna A. Bespalko;
- Formal analysis, preparing data for analysis, Leonid M. Naumchak;

- Research of satellites classification, Leonid M. Naumchak;
- Methodology for defining linguistic variables and terms, Iryna A. Bespalko and Leonid M. Naumchak;
- Project administration, Ihor A. Pilkevych;
- Software of modeling membership function, Iryna A. Bespalko;
- Supervision throughout the research process, Ihor A. Pilkevych and Dmytro V. Pekariev;
- Writing – original draft, Iryna A. Bespalko and Leonid M. Naumchak;
- Writing – review and editing, Iryna A. Bespalko.

References

[1] D. M. Vyporkhaniuk, S. V. Kovbasiuk, Basics of Space Situational Awareness. Foreign and domestic experience of space activities in security sector, Korolyov Zhytomyr Military Institute, 2018. URL: https://space.znau.edu.ua/images/book/monoghrafija2018.pdf.

[2] H. Azami, M.-R. Mosavi, S. Sanei, Classification of GPS Satellites Using Improved Back Propagation Training Algorithms, Wireless Personal Communications 71 (2013) 789–803. doi:10.1007/s11277-012-0844-7.

[3] O. Pronina, O. Piatykop, The recognition of speech defects using convolutional neural network, CTE Workshop Proceedings 10 (2023) 153–166. doi:10.55056/cte.554.

[4] S. O. Semerikov, T. A. Vakaliuk, I. S. Mintii, V. A. Hamaniuk, V. N. Soloviev, O. V. Bondarenko, P. P. Nechypurenko, S. V. Shokaliuk, N. V. Moiseienko, V. R. Ruban, Development of the computer vision system based on machine learning for educational purposes, Educational Dimension 5 (2021) 8–60. doi:10.31812/educdim.4717.

[5] R. C. Botelho A. S., A. L. Xavier Jr., A Unified Satellite Taxonomy Proposal Based on Mass and Size, Advances in Aerospace Science and Technology 4 (2019) 57–73. doi:10.4236/aast.2019.44005.

[6] Y. Huo, Z. Li, Y. Fang, F. Zhang, Classification for geosynchronous satellites with deep learning and multiple kernel learning, Appl. Opt. 58 (2019) 5830–5838. doi:10.1364/AO.58.005830.

[7] V. Omelchuk, D. Pekariev, O. Omelchuk, Generalization of the classification of space vehicles for remote sensing of the Earth, Bulletin of ZhSTU, Technical sciences 36 (2006) 75–80.

[8] I. Bespalko, V. Gerasimov, D. Pekariev, P. Piontkivskyi, R. Hryschuk, Analysis of space hyperspectral imaging systems features, Bulletin of ZhSTU, Technical sciences 60 (2012) 85–92. URL: https://eztuir.ztu.edu.ua/bitstream/handle/123456789/3285/14.pdf.

[9] I. Bespalko, D. Pekariev, A. Savchuk, R. Hryschuk, Taking into account the possibilities of space observation orbital means in the information provision of solving applied problems, Bulletin of ZhSTU, Technical sciences 64 (2013) 16–23. URL: https://library.ztu.edu.ua/e-copies/VISNUK/64_I/16.pdf.

[10] P. Friz, A method of selecting available spacecraft based on the conditions of geometric visibility between them and a given area of the Earth, Problems of creation, testing, application and operation of complex information systems 16 (2019) 94–107. doi:10.46972/2076-1546.2019.16.09.

[11] P. Friz, Software and modeling complex for informational decision-making support in tasks of space observations of the Earth, in: Problems of informatization: Proceedings of the ninth international scientific and technical conference, State University of Telecommunications, Kyiv, 2017, p. 6. URL: http://kist.ntu.edu.ua/konferencii/06_konf_2017.pdf.

[12] P. Friz, O. Kondratov, Algorithm for automated selection of relevant spacecraft for optical-electronic observation of given regions of the Earth, Bulletin of ZhSTU, Technical sciences 61 (2012) 138–146. URL: https://eztuir.ztu.edu.ua/bitstream/handle/123456789/3046/21.pdf.

[13] N. A. Antipova, O. I. Kulagin, Cloud resources on theory and methods of fuzzy sets and fuzzy logic, CTE Workshop Proceedings 2 (2014) 269–273. doi:10.55056/cte.217.

[14] I. M. Tsidylo, S. O. Semerikov, T. I. Gargula, H. V. Solonetska, Y. P. Zamora, A. V. Pikilnyak, Simulation of intellectual system for evaluation of multilevel test tasks on the basis of fuzzy logic, CTE Workshop Proceedings 8 (2021) 507–520. doi:10.55056/cte.304.

[15] A. V. Ryabko, O. V. Zaika, R. P. Kukharchuk, T. A. Vakaliuk, V. V. Osadchyi, Methods for predicting the assessment of the quality of educational programs and educational activities using a neuro-fuzzy approach, CTE Workshop Proceedings 9 (2022) 154–169. doi:10.55056/cte.112.

[16] O. Khorozov, Application of fuzzy logic for telemedicine systems, Kybernetyka y vychyslytelnaia tekhnyka 2 (2017) 36–48. doi:10.15407/kvt188.02.036.

[17] O. Pisarchuk, Modeling of situational management and identification processes in energetic information and management systems, Bulletin of ZhSTU, Technical sciences 71 (2014) 98–105. URL: http://eztuir.ztu.edu.ua/bitstream/handle/123456789/2197/17.pdf.

[18] T. Zheldak, L. Koryashkina, S. Us, Fuzzy sets in management and decision-making systems, National Technical University Dnipro Polytechnic, 2020. URL: https://ir.nmu.org.ua/bitstream/handle/123456789/156356/CD1239.pdf.

[19] Gunter's Space Page, Orbital Launches of 2023, 2023. URL: https://space.skyrocket.de/doc_chr/lau2023.htm#planned.

[20] Gunter's Space Page, Launch Vehicles, 2023. URL: https://space.skyrocket.de/directories/launcher.htm.

[21] European Space Agency, Types of orbits, 2023. URL: https://www.esa.int/Enabling_Support/Space_Transportation/Types_of_orbits.

Advanced software framework for comparing balancing strategies in container orchestration systems

Yevhenii V. Voievodin[1], Inna O. Rozlomii[1]

[1] *The Bohdan Khmelnytsky National University of Cherkasy, 81 Shevchenka Blvd., Cherkasy, 18031, Ukraine*

Abstract

This paper introduces a detailed software design for a system that evaluates scheduling strategies in container orchestration systems. Focusing on software architecture, it elaborates on the various components such as dynamic cluster topology and container configuration streams, cluster packing algorithms, metric collectors and a state machine for tracking experiment progress. The system incorporates malfunction scenarios, testing the resilience of different strategies. The system is designed to be flexible and open to be extended with new key performance indicators and test scenarios. The experiment flow is split into independent iterations that can be efficiently run in parallel enabling faster experiment executions. The paper reviews related work, positioning this system as an essential tool in the current research landscape for resource distribution and management in distributed systems. A key aspect of the design is the client-server architecture, which not only ensures scalability and adaptability for various experiments but also includes an API for enhanced interaction and result analysis. This comprehensive design approach makes the designed system a helpful tool for nuanced analysis and informed decision-making in container orchestration, with the potential to advance in the field by speeding up researches and creating a collection of strategy evaluation techniques.

Keywords

container orchestration systems, Kubernetes, Docker, Docker Swarm, software design, distributed systems, resources distribution

1. Introduction

Container orchestration systems (COS) are modern software that provide capabilities to maintain large and complex systems [1]. The primary technology that COS relies on is a container. Containers can be described as applications packed with all the dependencies they require, making the deployment of such applications easy and reproducible across different operating systems and platforms. The convenience of such deployments accelerates the development of applications [2].

There are multiple components that COS consists of, as illustrated in figure 1: a cluster containing nodes, with nodes containing containers, and also a scheduler. The scheduler decides which node to use for deploying the next container in the sequence. To do so, the scheduler uses a scheduling strategy. Typical strategies include “binpack” and “spread”, each aiming for different goals [3]. For example, “binpack” aims to maximize the utilization of nodes, while “spread” allows for better fault tolerance [4] of the deployed application.

The role of the strategy cannot be undervalued, but the choice of strategy is not easy to make. It depends on a variety of factors, such as resilience to failures, usage of resources, and resource locality. Machine learning is one direction where strategy development is heading, which makes the comparison process even more challenging [5].

The key goal of this article is to provide a comprehensive design for an application capable of evaluating the performance of two or more scheduling strategies. Such an application must be flexible enough to compare strategies regardless of their implementation and be easily extendable with new metrics and comparison techniques.

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine

yevhenii.voievodin@vu.cdu.edu.ua (Y. V. Voievodin); inna-roz@ukr.net (I. O. Rozlomii)
https://scholar.google.com/citations?user=AU5G-f0AAAAJ (Y. V. Voievodin);
https://scholar.google.com/citations?user=04ryMCwAAAAJ (I. O. Rozlomii)
0000-0002-6415-8566 (Y. V. Voievodin); 0000-0001-5065-9004 (I. O. Rozlomii)

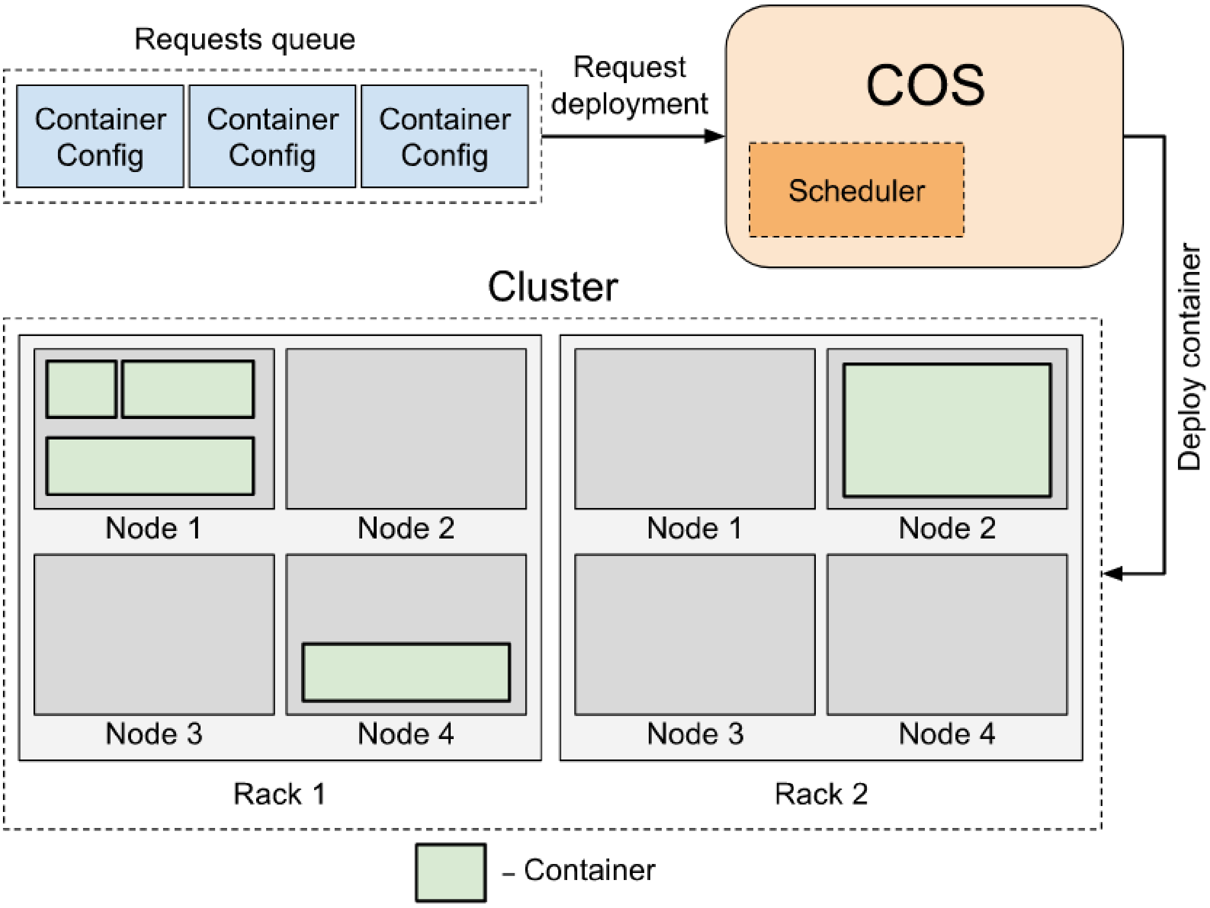


Figure 1: Key container orchestration system components.

2. Related works

The approach described by Voievodin et al. [6] to evaluate scheduling strategies provides a deep dive into the important role of the scheduling strategy choice. It proposes key performance indicators for comparing scheduling strategies and enlists ideas about which algorithms can be used to complete the evaluation from start to end. This includes packing algorithms, fault tolerance testing, and the aggregation of experiment results. This article delves deeper into the topic of strategy evaluation and proposes a more complete and sophisticated design for such evaluation software. While it builds on the proposed approach and algorithms, it extends the topic further by suggesting a concrete system structure and software techniques that can be used to implement such a system.

In the context of distributed systems, particularly those utilizing microservices architecture [7], COS serves as an essential tool, ensuring the efficient and easily scalable operation of independently deployed services across various computing environments at low overhead [8]. Saboor et al. [9] emphasize the importance of resource utilization in such applications, noting that they have gained rapid adoption in the software industry. A study on the aging and fault tolerance of microservices in Kubernetes provides useful insights on how COS, in the representation of Kubernetes, can achieve different fault tolerance properties and what options there are [10]. Gogouvitis et al. [11] discuss how container orchestration can be beneficial to seamless computing in industrial systems, which is software distributed across different computing domains. Akuthota [12] covers a technique of chaos engineering in distributed systems, which involves introducing controlled failures to the system to help make them more robust.

Many scheduling strategies have been developed recently, which higlights the actuality of the topic. An efficient virtual central processor unit scheduling in cloud computing [13]. Container scheduling using TOPSIS algorithm [14]. A combined priority scheduling method for distributed machine learning

[15]. A new container scheduling algorithm based on multi-objective optimization [16]. Improvement of container scheduling for docker using ant colony optimization [17]. A particle swarm optimization-based container scheduling algorithm of docker platform [18]. Contention-aware container placement strategy for docker swarm with machine learning based clustering algorithms [19].

3. Key components of the system

The experiment is a key component and consists of multiple parts, each of which must be separately configured. These parts include: a stream of configurations, a strategy, a packer, an iteration result collector, a malfunction algorithm, and a malfunction result collector. Different phases of the experiment are represented by its state. The state machine includes the states: NEW, RUNNING, COMPLETED, INTERRUPTED, and FAILED (figure 2).

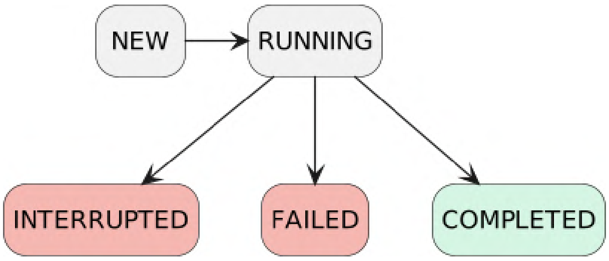


Figure 2: Experiment states and possible state transitions.

- NEW – indicates that the experiment can be configured. It has not been run yet, and new experiments might be incomplete in terms of configuration. Experiment components can be configured step by step.
- RUNNING – indicates that the experiment is currently running, which technically means going through the experiment flow (figure 3). Experiments that are running can no longer be modified in terms of configuration; the only change is that they accumulate data points for each new computed result.
- COMPLETED – indicates that the experiment has successfully completed. Such an experiment has run through all the configured steps and collected the desired metrics, which can now be analyzed.
- FAILED – indicates that the experiment was unable to complete successfully. This could be due to an unexpected error during execution or insufficient resources to finalize the experiment.
- INTERRUPTED – indicates that the execution of the experiment was deliberately interrupted. The reasons might vary, but primarily it could be to save resources when it’s clear from the results produced so far that no further executions are necessary.

The class diagram (figure 4) covers the key components of the experiment, offering a detailed look into interfaces and structures. Before the first iteration starts, there is a setup phase, as illustrated in figure 3b. The iteration setup includes the propagation of cluster topology to all the strategies. The topology essentially comprises a set of nodes that have limits and can contain deployed containers. This topology is generated by the topology stream (figure 3a), a crucial first step. The topology stream allows for the definition of virtually any cluster structure, including the placement of nodes in physical racks for further fault tolerance testing. Additionally, the topology stream determines when the experiment stops, as it concludes when there are no more topologies to run the experiment for. The generated topology then serves as a prototype [20] for subsequent experiment iterations.

The stream of configurations is responsible for generating container configurations (requirements) to be placed within a cluster. Firstly, the stream can be either finite or infinite. An infinite stream will continue generating configurations as long as the packer demands it. Finite streams, on the other hand,

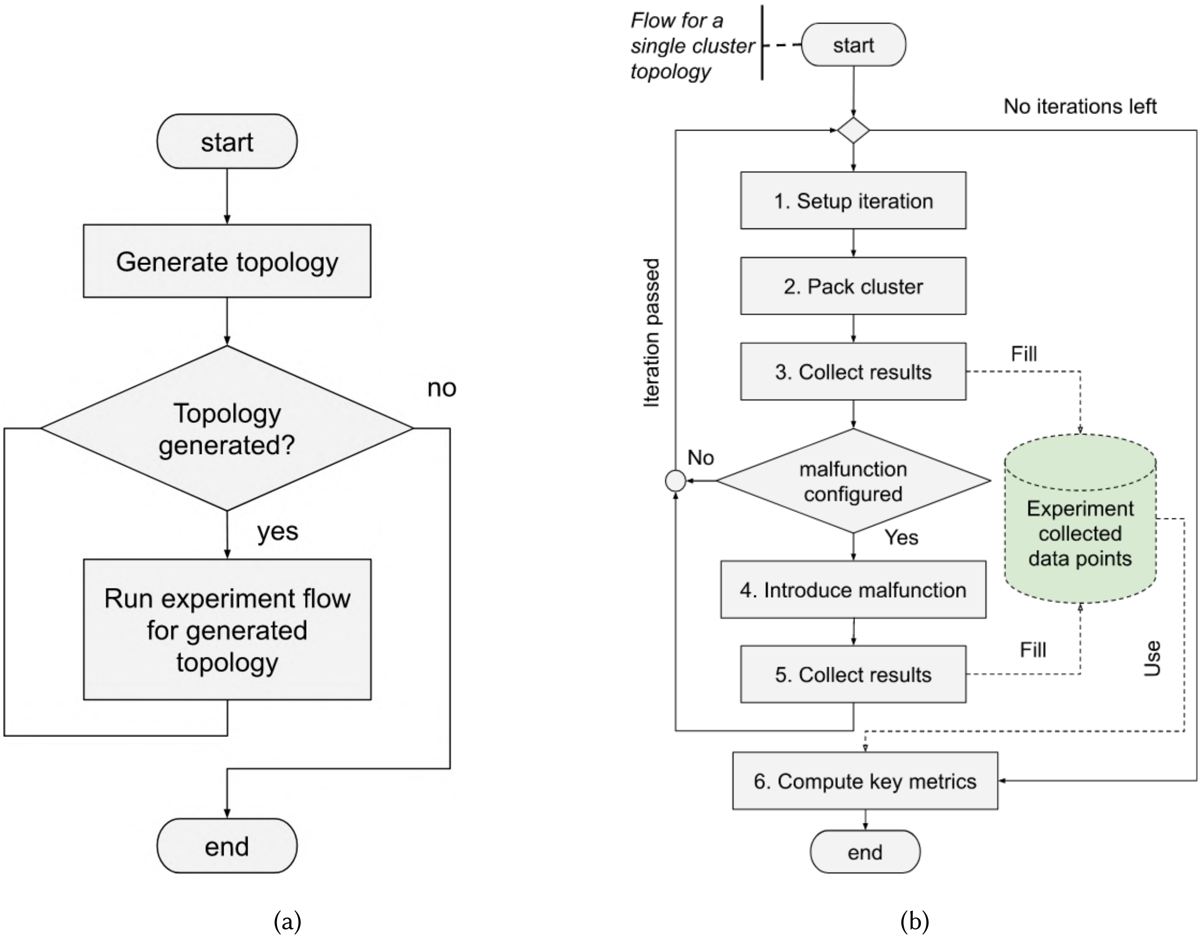


Figure 3: Experiment steps, (a) – topology generation, (b) – experiment flow for a single topology.

can be used to test scheduling strategies for a very specific set of container configurations, to seek a better strategy, or for strategy monitoring purposes. Regardless of whether a finite or infinite stream is used, virtually any sequence of containers can be provided, including random sequences. A crucial aspect is ensuring that the same sequence is fed to different strategies, where each strategy operates its own copy of the cluster to fill.

Secondly, the stream can be used to define application families. For instance, it can generate several configurations that depend on each other and form a larger application, as commonly seen in microservices architecture [21]. The specific implementation of the stream determines how to establish these dependencies, and the container configuration structure allows for such connections to be specified.

Thirdly, the stream can replicate a single container configuration multiple times, for example, if an application must be deployed multiple times within a cluster to ensure better response to failures [22]. Each aspect can be implemented as a separate stream representation. The decorator pattern [20] can be employed to combine these implementations in a desired manner, allowing for a variety of system demands to be covered.

The packer is tasked with making decisions regarding when to stop in the case of an infinite stream of configurations. As discussed by Voievodin et al. [6], such decisions are highly specific to the use-case being tested. For instance, the packer might stop after encountering the first scheduling request error, or once the cluster is full. While it is the packer's responsibility to execute scheduling algorithms, the packer itself does not depend on the specifics of the strategy implementation.

After the packing process is completed, the results aggregator collects the packing results based on the state of clusters filled by different strategies. Firstly, the aggregator's job is to collect important data points, which it does after the execution of every iteration. Secondly, it produces an aggregate that

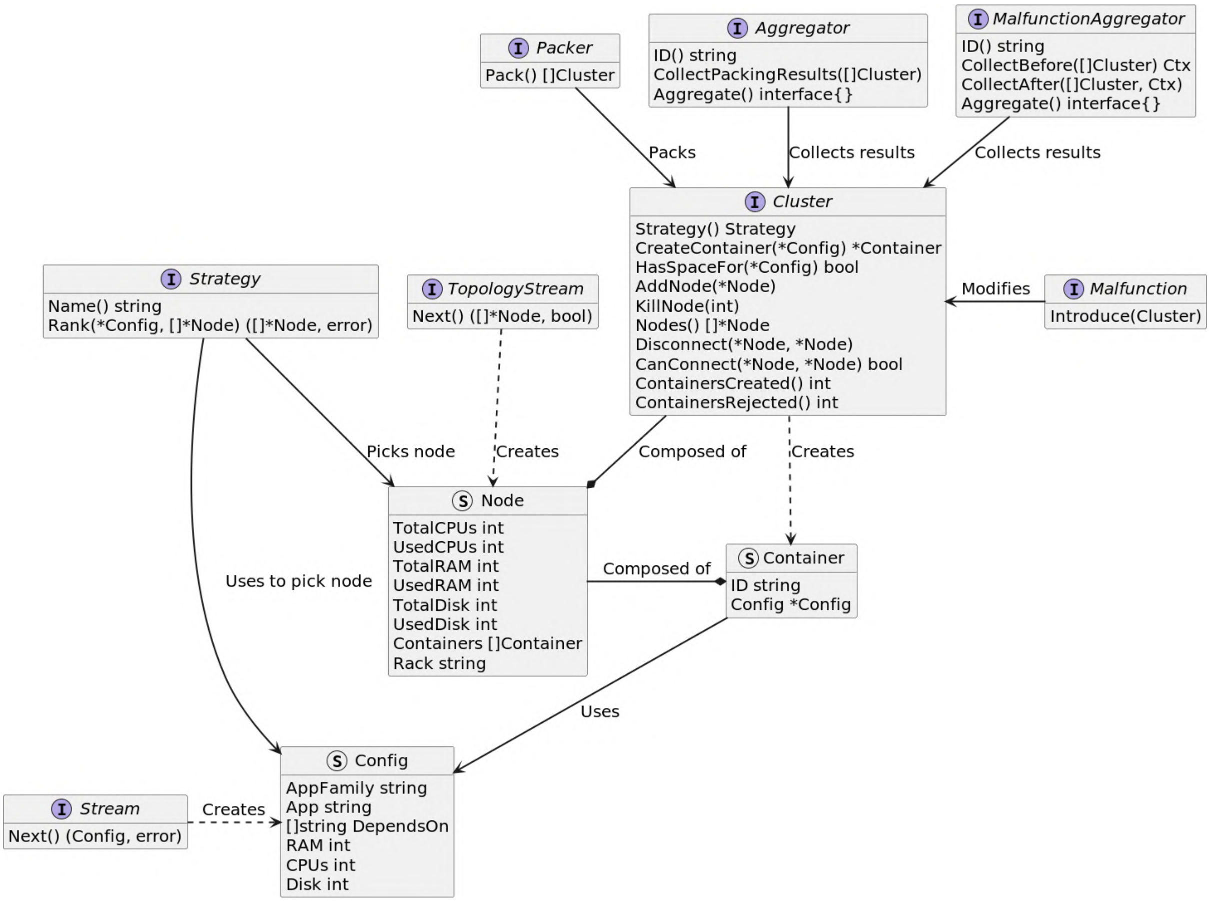


Figure 4: Class diagram of key system components (types notion is Golang specific).

represents these data points. For example, the aggregator might count the number of containers deployed by each strategy and then compute the average number of containers deployed. The aggregation phase occurs after the last iteration for the current topology has been executed. The flexible interface of the results aggregator allows for the production of a wide range of statistical information. For instance, with all the data points collected, an aggregate might include percentile or median values. The aggregated results are stored within the experiment and get associated with the corresponding topology.

One of the desired characteristics of a strategy is its management of the fault tolerance of the deployed system. The "malfunction" component assists in testing this aspect [23]. It's important to describe the malfunction in combination with the malfunction results collector. This collector is similar to the previously described collector, with the primary difference being that it collects results twice: before and after the malfunction is introduced. This approach enables the malfunction results collector to compare changes resulting from the operation of the malfunction algorithm. For instance, it can assess how many applications or application families survived a network partition [24]. The malfunction operates within the cluster, deliberately causing a disruption, such as removing a node from the cluster (figure 5) or reducing the percentage of available connections. Since everything is interconnected by default, the cluster provides a means to disconnect two nodes.

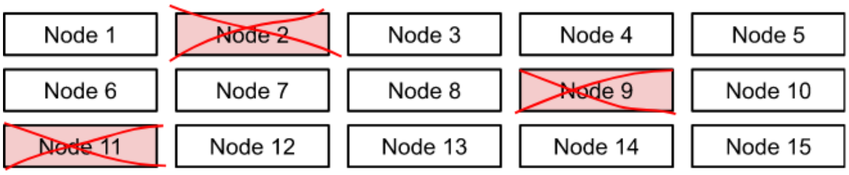


Figure 5: Example of malfunction removing random nodes in the cluster.

4. Parallel execution

The organization of the flow facilitates faster experiment execution in multiprocessor systems [25]. Each experiment iteration is executed in a separate thread, which accelerates the overall experiment execution speed, especially since most of the work occurs in the packer. To effectively collect results between iterations, proper synchronization techniques must be employed. In this context, a common Mutex implementation will suffice. Within the scope of a single experiment, it definitely makes sense to parallelize both iterations and topologies, as they can be executed independently of each other.

Furthermore, the application architecture allows for a higher level of parallelization, presenting additional opportunities. The system can be utilized to identify the most suitable topology for a given sequence of containers. This can be achieved by executing different experiments, each with a distinct topology stream. Virtually any configuration or additional testing techniques can be applied at this higher level, utilizing the existing experiment mechanics. Since each experiment is self-contained, these algorithms can also be parallelized.

5. High level organisation of the system

A client-server architecture [19] is a recommended choice for such an application. Firstly, the implementation of the previously described components is separate from the visualization of the experiment results. The system's flexible state allows for the choice of whether to represent such results with user interface components, or whether another system should simply delegate the execution of experiments to this one while making decisions based on the experiment results. Another advantage of adopting a client-server architecture is the ability to have multiple server instances, thereby enabling high-level parallelization of experiment execution. Additionally, having multiple server instances enhances the overall resilience of the system.

The server component of the system must expose an application programming interface (API) to utilize the previously described features (figure 6). Modern client-server systems typically use REST API [26] or gRPC [27]. The API functions of the proposed software are straightforward and can be implemented using the most preferred approaches. These functions include:

- Create experiment: This function creates a new experiment with all default values set, which cannot be run yet. The created experiment will be in the NEW state.
- Update experiment: This method adds new configurations to the experiment. It allows for step-by-step configuration of the experiment, cloning of experiments, and modification of their parts. It technically facilitates quick testing of various hypotheses.
- Find experiments by id or other attributes: This function enables the retrieval of information on previously run experiments and their results, as the results are part of the experiment data.
- Clone experiment: This creates an identical clone of an experiment, which can then be modified to observe different behaviors. With many configuration options available, it makes sense to change some dimensions and observe how the results vary. For example, adding a new strategy to the list of tested ones and observing the impact on results.
- Execute experiment: This starts the experiment execution, transitioning the experiment to the EXECUTING state and commencing the broadcasting of all previously described events.
- Interrupt experiment execution: This stops the experiment execution and transitions the experiment to the INTERRUPTED state. In cases where it becomes apparent that the experiment is not yielding expected results, continuing the execution would be unproductive and consume more resources. The experiment can thus be interrupted to conserve resources.
- Subscribe to experiment events: Once a subscription is made, the subscriber will receive all the events of interest.

The state of the experiment encompasses all the experiment results, even if the experiment is currently running. Since intermediate results are still useful, they must be distributed over the API to interested consumers. These events include:

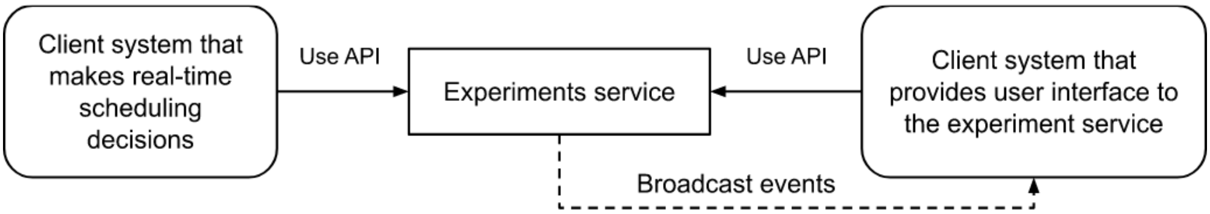

Client system that makes real-time scheduling decisions
Use API
Experiments service
Use API
Client system that provides user interface to the experiment service
Broadcast events

Figure 6: Example of two clients and one server instance.

- Experiment state changes.
- Experiment iteration completion. This event can be throttled to avoid overwhelming the client system.
- Experiment result available. Sent every time the experiment is executed for one of the cluster topologies, indicating that there is a new experiment result entry available, which can then be represented on the client side.
- Experiment execution for a topology started. This event is purely technical and ensures that the client side accurately displays the necessary progress.

Internally, broadcasting is implemented following the event listener pattern [20], where the system's role is to send the event to interested subscribers. Externally, these events will be broadcast over the network to connected clients, enabling them to make quick decisions regarding the progress of experiment execution.

6. Interpretation of experiment results

The aim of results interpretation is to determine which strategy performs better or worse in certain scenarios. Charts and tables are ideal tools for illustrating such comparisons. The comparison itself is based on the experiment results, which include values produced by the aggregators. Each set of values has an identification that allows for differentiating between the aggregates.

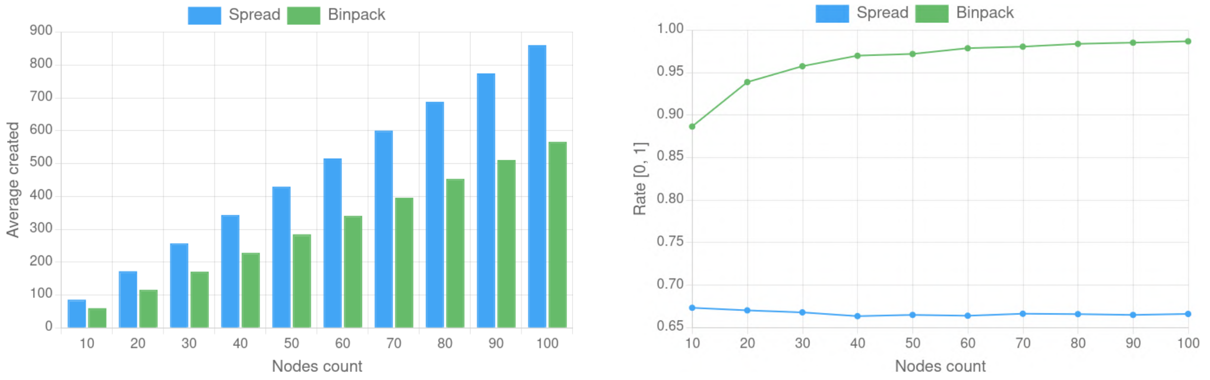
For instance, suppose an aggregator computes the average number of containers created by different strategies. The results are then represented as the average number of containers per strategy for each topology. To analyze these results, a histogram chart can be used [28]. An example histogram (figure 7a) might clearly indicate that, for all topologies, the binpack strategy managed to create fewer containers on average before the packing condition was met in this particular experiment setup.

Additionally, rates, such as the container creation rate (the percentage of successfully satisfied scheduling requests), can be displayed using a line chart. In an example (figure 7b), the "binpack" strategy may be shown to reject significantly fewer container scheduling requests compared to the "spread" strategy.

7. Discussion

While this article comprehensively covers the design of the application, it is important to remember that this is not the software itself. The choice of technology and the discussion around the alternative higher-level organization of components remain open topics. A judicious selection of technology and supporting infrastructure is crucial to ensure that the designed software remains both flexible and scalable. One effective approach to organize such a project is to make it open source, thus allowing contributions from all interested parties.

Another promising direction for this system is the development of a real-time system that relies on experiment results to make further scheduling decisions. It's also important to ensure that the system can be extended with new key performance indicators and strategy algorithms. A potential next step



(a)

(b)

Figure 7: Example of charts used to represent experiment results, (a) – average created containers, (b) – container creation rate.

could be exploring the most suitable structure for the cluster, rather than just looking for a strategy that fits a certain setup. By doing so, the experiment could provide even more valuable insights.

8. Conclusion

This paper delves into the design of a system that enables the evaluation of scheduling strategy algorithms within the context of distributed systems, particularly in relation to microservices architectures where COS is extensively utilized. The related works underline the importance of selecting the appropriate strategy and show how such a decision could affect different parts of distributed systems, like resource utilization or fault tolerance.

One of the main goals when designing such a system is to ensure its flexibility. This flexibility allows for the testing of different aspects of distributed systems reliant on COS. The proposed division of responsibilities among different components, such as the topology stream, configuration stream, packer, strategy, cluster, nodes, containers, aggregators, and malfunctions, allows for extensive customization of the experiment flow to achieve the desired behavior. For example, such system can be used to compare the degree of resource fragmentation on the cluster nodes and thus assess the efficiency of resource utilization, measure the rates of containers creation or rejection, evaluate the availability of applications encountering various network partitions or node failures. The client-server organization of these components separates the representation of results from the experiment execution itself. This separation removes any assumptions about how experiment results can be utilized, thereby opening up a variety of other use cases, such as enabling a higher-level system that relies on the experiment's API for making scheduling decisions.

The next step would be the implementation of such a system. This could significantly accelerate further research in the fields of resource distribution and distributed systems. The system would not only offer a platform for experimentation but also become a valuable source of knowledge about scheduling algorithms.

9. Authors contribution

The authors confirm contribution to the paper as follows: study conception and design: Voievodin Y., Rozlomii I.; data collection: Voievodin Y.; analysis and interpretation of results: Voievodin Y., Rozlomii I.; manuscript preparation: Voievodin Y. All authors reviewed the results and approved the final version of the manuscript.

References

[1] M. A. Rodriguez, R. Buyya, Container-based cluster orchestration systems: A taxonomy and future directions, Software: Practice and Experience 49 (2019) 698–719. doi:https://doi.org/10.1002/spe.2660.

[2] T. Siddiqui, S. A. Siddiqui, N. A. Khan, Comprehensive Analysis of Container Technology, in: 2019 4th International Conference on Information Systems and Computer Networks (ISCON), 2019, pp. 218–223. doi:10.1109/ISCON47742.2019.9036238.

[3] H. M. Fard, R. Prodan, F. Wolf, Dynamic Multi-objective Scheduling of Microservices in the Cloud, in: 2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC), 2020, pp. 386–393. doi:10.1109/UCC48980.2020.00061.

[4] P. Kumari, P. Kaur, A survey of fault tolerance in cloud computing, Journal of King Saud University - Computer and Information Sciences 33 (2021) 1159–1176. doi:10.1016/j.jksuci.2018.09.021.

[5] Z. Zhong, M. Xu, M. A. Rodriguez, C. Xu, R. Buyya, Machine Learning-based Orchestration of Containers: A Taxonomy and Future Directions, ACM Comput. Surv. 54 (2022) 217. doi:10.1145/3510415.

[6] Y. Voievodin, I. Rozlomii, A. Yarmilko, Approach to Evaluate Scheduling Strategies in Container Orchestration Systems, in: Modeling, Control and Information Technologies: Proceedings of International scientific and practical conference, 6, 2023, pp. 292–295. doi:10.31713/mcit.2023.089.

[7] V. Bushong, A. S. Abdelfattah, A. A. Maruf, D. Das, A. Lehman, E. Jaroszewski, M. Coffey, T. Cerny, K. Frajtak, P. Tisnovsky, M. Bures, On Microservice Analysis and Architecture Evolution: A Systematic Mapping Study, Applied Sciences 11 (2021) 7856. doi:10.3390/app11177856.

[8] I. M. A. Jawarneh, P. Bellavista, F. Bosi, L. Foschini, G. Martuscelli, R. Montanari, A. Palopoli, Container Orchestration Engines: A Thorough Functional and Performance Comparison, in: ICC 2019 - 2019 IEEE International Conference on Communications (ICC), 2019, pp. 1–6. doi:10.1109/ICC.2019.8762053.

[9] A. Saboor, M. F. Hassan, R. Akbar, S. N. M. Shah, F. Hassan, S. A. Magsi, M. A. Siddiqui, Containerized Microservices Orchestration and Provisioning in Cloud Computing: A Conceptual Framework and Future Perspectives, Applied Sciences 12 (2022) 5793. doi:10.3390/app12125793.

[10] J. Flora, P. Gonçalves, M. Teixeira, N. Antunes, A Study on the Aging and Fault Tolerance of Microservices in Kubernetes, IEEE Access 10 (2022) 132786–132799. doi:10.1109/ACCESS.2022.3231191.

[11] S. V. Gogouvitis, H. Mueller, S. Premnadh, A. Seitz, B. Bruegge, Seamless computing in industrial systems using container orchestration, Future Generation Computer Systems 109 (2020) 678–688. doi:10.1016/j.future.2018.07.033.

[12] A. Akuthota, Chaos Engineering for Microservices, A Starred Paper Submitted in Partial Fulfillment of the Requirements for the Degree Master of Science in Computer Science, St. Cloud State University, 2023. URL: https://repository.stcloudstate.edu/csit_etds/42/.

[13] J. Jang, J. Jung, J. Hong, An efficient virtual cpu scheduling in cloud computing, Soft Computing 24 (2020) 5987–5997. doi:10.1007/s00500-019-04551-w.

[14] A. P. Shriniwar, Container Scheduling Using TOPSIS Algorithm, Msc research project, National College of Ireland, Dublin, 2020. URL: https://norma.ncirl.ie/4551/.

[15] T. Du, G. Xiao, J. Chen, C. Zhang, H. Sun, W. Li, Y. Geng, A combined priority scheduling method for distributed machine learning, EURASIP Journal on Wireless Communications and Networking 2023 (2023) 45. doi:10.1186/s13638-023-02253-4.

[16] B. Liu, P. Li, W. Lin, N. Shu, Y. Li, V. Chang, A new container scheduling algorithm based on multi-objective optimization, Soft Computing 22 (2018) 7741–7752. doi:10.1007/s00500-018-3403-7.

[17] C. Kaewkasi, K. Chuenmuneewong, Improvement of container scheduling for Docker using Ant Colony Optimization, in: 2017 9th International Conference on Knowledge and Smart Technology

(KST), 2017, pp. 254–259. doi:10.1109/KST.2017.7886112.

[18] L. Li, J. Chen, W. Yan, A particle swarm optimization-based container scheduling algorithm of docker platform, in: Proceedings of the 4th International Conference on Communication and Information Processing, ICCIP '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 12–17. doi:10.1145/3290420.3290432.

[19] S. A. Hamid, R. A. Abdulrahman, R. A. Khamees, What is Client-Server System: Architecture, Issues and Challenge of Client-Server System, Recent Trends in Cloud Computing and Web Engineering 2 (2020) 1–6. doi:10.5281/zenodo.3673071.

[20] E. Freeman, E. Robson, Head First Design Patterns, O'Reilly Media, 2020. URL: https://github.com/ajitpal/BookBank/blob/master/%5BO%60Reilly.%20Head%20First%5D%20-%20Head%20First%20Design%20Patterns%20-%20%5BFreeman%5D.pdf.

[21] C. Surianarayanan, G. Ganapathy, R. Pethuru, Essentials of Microservices Architecture Paradigms, Applications, and Techniques, Taylor & Francis, 2019.

[22] L. Abdollahi Vayghan, M. A. Saied, M. Toeroe, F. Khendek, Deploying Microservice Based Applications with Kubernetes: Experiments and Lessons Learned, in: 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), 2018, pp. 970–973. doi:10.1109/CLOUD.2018.00148.

[23] J. Bergstrom, Chaos Engineering, The ITEA (2022) 208. URL: https://itea.org/the-itea-journal/journal-at-a-glance/.

[24] M. Kleppmann, Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems, O'Reilly Media, 2017.

[25] M. Herlihy, N. Shavit, V. Luchangco, M. Spear, The Art of Multiprocessor Programming, Newnes, 2020. URL: https://cs.ipm.ac.ir/asoc2016/Resources/Theartofmulticore.pdf.

[26] H. Subramanian, P. Raj, Hands-On RESTful API Design Patterns and Best Practices: Design, develop, and deploy highly adaptable, scalable, and secure RESTful web APIs, Packt Publishing Ltd, 2019.

[27] K. Indrasiri, D. Kuruppu, gRPC: Up and Running: Building Cloud Native Applications with Go and Java for Docker and Kubernetes, O'Reilly Media, 2020.

[28] S. D. H. Evergreen, Effective Data Visualization: The Right Chart for the Right Data, 2 ed., SAGE publications, 2019.

Data serialization protocols in IoT: problems and solutions using the ThingsBoard platform as an example

Dmytro I. Shvaika[1,2], Andrii I. Shvaika[1,2] and Volodymyr O. Artemchuk[1,3,4,5]

[1]*G.E. Pukhov Institute for Modelling in Energy Engineering of the National Academy of Sciences of Ukraine, 15 General Naumov Str., Kyiv, 03164, Ukraine*

[2]*ThingsBoard, Inc., 110 Duane Street, Suite 1C, New York, 10007, USA*

[3]*Center for Information-analytical and Technical Support of Nuclear Power Facilities Monitoring of the NAS of Ukraine, 34a Palladin Ave., Kyiv, 03142, Ukraine*

[4]*Kyiv National Economic University named after Vadym Hetman, 54/1 Peremohy Ave., Kyiv, 03057, Ukraine*

[5]*National Aviation University, 1 Liubomyra Huzara Ave., Kyiv, 03058, Ukraine*

Abstract

This article delves into the challenges and advancements in data serialization protocols within the Internet of Things (IoT), primarily focusing on dynamic schema compilation in ThingsBoard. A comparative analysis of Protobuf against other serialization protocols like JSON, XML, and PSON highlights Protobuf's efficiency and outlines the necessity for flexible ways of device integration that use Protocol Buffers for data transmission. We identify the limitations of static schema compilation in Protobuf and propose a novel approach for real-time, user-driven schema compilation that enhances flexibility, scalability, and performance in IoT platforms. Our solution addresses critical adaptability issues by enabling seamless device communication and integration using compact Protobuf formats. We emphasize the potential impact of this solution in the scope of edge computing and suggest directions for future research to broaden the applicability of dynamic serialization across various IoT solutions. This work contributes to improving IoT data management and paves the way for more adaptable and efficient IoT ecosystems.

Keywords

IoT Platform, Data Serialization, Protocol Buffers, ThingsBoard

1. Introduction

In the contemporary landscape, where the Internet of Things (IoT) is gaining prominence [1], data processing and transmission effectiveness emerge as a pivotal determinant of technological success. Data serialization protocols play a crucial role in this domain, facilitating the exchange of information among IoT devices in a compact and efficient format. Widely employed protocols like JSON, XML, Protocol Buffers, and others cater to various IoT systems, addressing the demand for swift and dependable communication. Nevertheless, each protocol presents unique challenges and constraints concerning integration and scalability within intricate IoT ecosystems.

The Internet of Things (IoT) is a rapidly evolving field with many applications. Debnath and Chettri [2] and Villamil et al. [3] highlight IoT's diverse applications, including in industry, business, and improving quality of life. Uckelmann et al. [4] emphasizes the potential for IoT to revolutionize business processes and enable a more convenient way of life. Porkodi and Bhuvaneswari [5] provides a detailed overview of the communication-enabling technology standards in IoT, such as RFID tags and sensors. The study by Khang et al. [6] addresses the limitations of single-path communication in hydroponic systems, emphasizing the need for reliable multi-path communication in IoT-based monitoring systems.

However, when it comes to the specific topic of data serialization protocols in IoT, the literature is relatively scarce (Luis et al. [7], Friesel and Spinczyk [8], Domínguez-Bolaño et al. [9], Pustišek

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine

shvaikad@gmail.com (D. I. Shvaika); andrew.shvayka@gmail.com (A. I. Shvaika); ak24avo@gmail.com (V. O. Artemchuk)
https://www.researchgate.net/profile/Volodymyr-Artemchuk-2 (V. O. Artemchuk)
0009-0001-3088-3997 (D. I. Shvaika); 0009-0006-8461-3550 (A. I. Shvaika); 0000-0001-8819-4564 (V. O. Artemchuk)

et al. [10], Delgado [11], Jacoby and Usländer [12], Deniziak et al. [13], Jiang et al. [14], Hou et al. [15], Hasemann et al. [16], Kolbe et al. [17], Kharat et al. [18], Khodadadi and Sinnott [19]).

The ThingsBoard Platform has garnered substantial popularity among researchers, as evidenced by numerous publications dedicated to its utilization. In particular, the following examples highlight its prominence in the academic community (Ilyas et al. [20], Henschke et al. [21], Aghenta and Iqbal [22], De Paolis et al. [23], Casillo et al. [24], Okhovat and Bauer [25], Bestari and Wibowo [26], Sabuncu and Thornton [27], Jang et al. [28], Kadarina and Priambodo [29]).

In this article, we focus on analyzing data serialization protocols within the context of IoT, examining their applications and the challenges they present to developers and engineers. The ThingsBoard platform, recognized as one of the leading open-source IoT platforms, is a practical instrument in this investigation, allowing for a detailed analysis of various facets of data serialization. Its adaptability and scalability in addressing IoT device management and data processing tasks make it an ideal candidate for delving into the intricacies of serialization protocols within IoT environments.

Research object is data serialization protocols in distributed IoT systems, emphasizing utilizing methods and mechanisms for data transfer between devices and the system. *Research subject* is characteristics and performance of serialization protocols, encompassing data size, processing speed, and utilizing ThingsBoard for practical analysis. *Research objective* is to analyze and assess data serialization protocols in the IoT landscape, delineating their advantages and exploring avenues for improvement, specifically focusing on their impact on performance and flexibility across various IoT scenarios.

2. Comparative analysis of data serialization protocols for IoT

An ordinary device transforms into an IoT device upon integration with an IoT platform, functioning through data exchange with fellow IoT devices or cloud servers. This necessitates a standardized data exchange format at the application level. To address this challenge, libraries offering standardized data formats are readily accessible. However, the costs related to data (de)serialization and transmission with these libraries are largely undocumented in the realm of IoT, or documented in limited capacities for specific protocols. The Friesel and Spinczyk [8] study examined JSON JSON [30] encoding efficiency within the IoT framework. This involved a comparative analysis juxtaposing JSON with alternative serialization formats. The results underscored the efficacy of Protocol Buffers, or Protobuf, highlighting their suitability for energy-efficient data serialization in the context of contemporary, high-capacity IoT devices. The Luis et al. [7] study focused on assessing the performance metrics of PSON, comparing it against a spectrum of formats, including Protocol Buffers Google [31]. This comprehensive analysis covered various dimensions, such as serialization/deserialization velocities, binary file dimensions, and encoding sizes. Building upon the findings of these studies, we present a comparative analysis tailored to elucidate the strengths and limitations of these protocols within the context of IoT applications. The table 1 provide key characteristics of leading data serialization protocols, highlighting their respective advantages and constraints.

Table 1
Comparative analysis of data serialization protocols for IoT (based on Luis et al. [7], Friesel and Spinczyk [8]).

Feature	Protobuf	JSON	XML	PSON
Format type	Binary	Text-based	Text-based	Binary
Efficiency (size)	High	Medium	Low	High
Efficiency (speed)	High	Medium	Low	High
Human readable	No	Yes	Yes	No
Language support	High	High	High	Medium
Extensibility	Yes	Yes	Yes	Yes
Versioning support	Yes(Proto2, Proto3)	No	No	No
IoT device compatibility	High	High	Medium	High

It is clear from the benchmarking that Protobuf is the leader in data serialization for IoT due to its high efficiency in both size and speed, wide language support and extensibility. Despite the rapid development and potential advantages of formats such as PSON, the presence of Protobuf and its continued use in various IoT applications reaffirms its importance.

3. Challenge of device integration over Protocol Buffers in IoT platforms

The challenge of integrating devices over Protocol Buffers in IoT platforms is a universal issue, not confined to a specific platform. Consequently, for our analysis, we've chosen ThingsBoard as our research tool. ThingsBoard, Inc. was founded in 2016 by a team of programmers from Ukraine and specializes in the development of software products for the IoT. ThingsBoard [32], with its open-source nature and comprehensive features, provides a robust foundation for exploring these challenges and potential solutions in a detailed and practical manner.

The IoT developers at ThingsBoard opted for schemaless JSON formats for primary serialization in external communication, facilitating data exchange with IoT devices due to their user friendly nature. In the ThingsBoard system, Protocol Buffers is used for inter-component data exchange. This decision is motivated by the need for streamlined processing of substantial data volumes while maintaining superior system performance. The compact nature and rapid serialization/deserialization of Protocol Buffers render it an optimal selection for enhancing internal network efficiency.

Currently, there is a growing interest in utilizing Protocol Buffers directly at the device level. Certain IoT devices transmit data solely through Protocol Buffers, while other users seek ways to transition to this format to enhance efficiency and reduce network load.

The integration of IoT devices that exclusively communicate using Protocol Buffers into IoT platforms exemplifies a pressing challenge, particularly for open-source platforms like ThingsBoard. Protobuf's static nature necessitates additional developer intervention for each new device type, undermining the platform's universality and scalability, especially in cloud deployments. To integrate a new Protobuf-compatible device, developers must manually define and compile the device's schema into the platform's codebase. This process that is both time-consuming and prone to errors.

A notable example is the integration of Efento devices into ThingsBoard using CoAP and Protobuf for seamless connectivity. The Efento [33] describes the interaction between Efento NB-IoT sensors and the ThingsBoard platform. Simultaneously, with device firmware versions in constant evolution, a scenario emerges wherein the platform must continually adapt to support new or updated devices. This interdependence raises questions about the sustainability of the platform in the IoT environment.

This scenario underscores the necessity for IoT platforms to develop more dynamic and versatile data serialization solutions. A mechanism that allows for the real-time, dynamic compilation and loading of Protobuf schemas would revolutionize device integration, enabling seamless adaptation to new devices and data formats without extensive developer intervention or system disruption.

4. Dynamic schema compilation in Protobuf by ThingsBoard

The preference of Protocol Buffers in IoT applications lies in its binary format's efficiency and the reduced load it imposes on network transmission. However, its static nature presents a formidable challenge. Typically, .proto files must be pre-compiled using the Protobuf compiler (protoc), producing source code for the desired programming languages. Any alterations to the schema necessitate a tedious cycle of recompilation and redeployment, impeding the rapid adaptability required in the fluid IoT ecosystems.

Addressing this, we propose a software tool enabling the real-time compilation of user-uploaded Protobuf schemas. This approach departs from traditional methods by allowing dynamic interpretation of Protobuf schema, thus permitting devices to communicate their data in Protobuf without necessitating system downtime or recompilation of the entire codebase. The solution is encapsulated within the

ThingsBoard platform through the concept of Device Profiles [34], which associate devices with their respective data transmission schemas.

In practice, each schema represents a distinct device's communication blueprint. Once a device is authenticated, its linked profile helps identify the pertinent schema for message interpretation. This dynamic process significantly lightens network traffic, as data is transmitted in Protobuf's compact form and only translated into a more verbose format like JSON when user interaction or specific system functions necessitate it.

This approach ensures that as IoT devices evolve or new ones join the network, the system can swiftly accommodate them without extensive manual interventions or halts in operation. It represents a leap toward an adaptable IoT platform capable of keeping pace with the sector's rapid growth and the diverse array of devices it encompasses.

5. Conclusions

This article explored the evolving landscape of data serialization protocols in IoT, with a special focus on the dynamic schema compilation feature within ThingsBoard. We've demonstrated how Protobuf, despite its efficiency and reduced network load, faces challenges in static schema compilation, limiting IoT devices' adaptability. Our findings suggest that the innovative solution of real-time, user-driven schema compilation can significantly enhance IoT platforms' flexibility, scalability, and overall performance. By enabling devices to communicate using compact Protobuf formats while allowing for seamless integration of new or updated devices, this approach addresses key scalability and adaptability challenges.

For future research and development, it would be insightful to delve deeper into how such dynamic data serialization mechanisms can further benefit edge computing scenarios. Specifically, investigating the impact on latency reduction, bandwidth optimization, and overall system responsiveness when deploying IoT devices in edge-centric networks. Additionally, exploring the integration of these serialization techniques with edge computing models could offer novel approaches to managing data flow and processing between edge devices and central systems, ultimately contributing to the scalability and robustness of IoT solutions.

References

[1] Y. B. Shapovalov, Z. I. Bilyk, S. A. Usenko, V. B. Shapovalov, K. H. Postova, S. O. Zhadan, P. D. Antonenko, Harnessing personal smart tools for enhanced STEM education: exploring IoT integration, Educational Technology Quarterly 2023 (2023) 210–232. doi:10.55056/etq.604.

[2] D. Debnath, S. K. Chettri, Internet of Things: Current Research, Challenges, Trends and Applications, in: X.-Z. Gao, R. Kumar, S. Srivastava, B. P. Soni (Eds.), Applications of Artificial Intelligence in Engineering, Algorithms for Intelligent Systems, Springer, Singapore, 2021, pp. 679–694. doi:978-981-33-4604-8_52.

[3] S. Villamil, C. Hernandez, G. Tarazona, An overview of internet of things, TELKOMNIKA (Telecommunication Computing Electronics and Control) 18 (2020) 2320–2327. doi:10.12928/telkomnika.v18i5.15911.

[4] D. Uckelmann, M. Harrison, F. Michahelles (Eds.), Architecting the Internet of Things, Springer, Berlin, Heidelberg, 2011. doi:10.1007/978-3-642-19157-2.

[5] R. Porkodi, V. Bhuvaneswari, The Internet of Things (IoT) Applications and Communication Enabling Technology Standards: An Overview, in: 2014 International Conference on Intelligent Computing Applications, Coimbatore, India, 2014, pp. 324–329. doi:10.1109/ICICA.2014.73.

[6] A. W. Y. Khang, J. A. J. Alsayaydeh, J. A. B. M. Gani, J. B. Pusppanathan, A. A. Teh, A. F. M. F. Ismail, T. K. Geok, Reliable Multi-Path Communication for IoT Based Solar Automated Monitoring as Motivation Towards Multi-Farming Hydroponic, International Journal of Interactive Mobile Technologies 17 (2023) 115–128. doi:10.3991/ijim.v17i21.43555.

[7] Á. Luis, P. Casares, J. J. Cuadrado-Gallego, M. A. Patricio, PSON: A Serialization Format for IoT Sensor Networks, Sensors 21 (2021) 4559. doi:10.3390/s21134559.

[8] D. Friesel, O. Spinczyk, Data Serialization Formats for the Internet of Things, in: Conference on Networked Systems 2021 (NetSys 2021), Electronic Communications of the EASST, Berlin, 2021. URL: https://journal.ub.tu-berlin.de/eceasst/article/view/1134/1078. doi:10.14279/tuj.eceasst.80.

[9] T. Domínguez-Bolaño, O. Campos, V. Barral, C. J. Escudero, J. A. García-Naya, An overview of IoT architectures, technologies, and existing open-source projects, Internet of Things 20 (2022) 100626. doi:10.1016/j.iot.2022.100626.

[10] M. Pustišek, A. Umek, A. Kos, Approaching the communication constraints of ethereum-based decentralized applications, Sensors 19 (2019) 2647. doi:10.3390/s19112647.

[11] J. C. M. Delgado, An Interoperability Framework and Distributed Platform for Fast Data Applications, in: Z. Mahmood (Ed.), Data Science and Big Data Computing: Frameworks and Methodologies, Springer International Publishing, Cham, 2016, pp. 3–39. doi:10.1007/978-3-319-31861-5_1.

[12] M. Jacoby, T. Usländer, Digital twin and internet of things-Current standards landscape, Applied Sciences 10 (2020) 6519. doi:10.3390/APP10186519.

[13] S. Deniziak, M. Płaza, Ł. Arcab, Approach for Designing Real-Time IoT Systems, Electronics 11 (2022) 4120. doi:10.3390/electronics11244120.

[14] T. Jiang, X. Huang, S. Song, C. Wang, J. Wang, R. Li, J. Sun, Non-Blocking Raft for High Throughput IoT Data, in: Proceedings - International Conference on Data Engineering, volume 2023-April, 2023, pp. 1140–1152. doi:10.1109/ICDE55515.2023.00092.

[15] C.-D. Hou, D. Li, J.-F. Qiu, L. Cui, EasiDEF: a horizontal lightweight data exchange protocol for internet of things, Jisuanji Xuebao/Chinese Journal of Computers 38 (2015) 602–613. doi:10.3724/SP.J.1016.2015.00602.

[16] H. Hasemann, A. Kröller, M. Pagel, RDF provisioning for the Internet of Things, in: 2012 3rd IEEE International Conference on the Internet of Things, 2012, pp. 143–150. doi:10.1109/IOT.2012.6402316.

[17] N. Kolbe, J. Robert, S. Kubler, Y. L. Traon, PROFICIENT: Productivity Tool for Semantic Interoperability in an Open IoT Ecosystem, in: Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, MobiQuitous 2017, Association for Computing Machinery, New York, NY, USA, 2017, p. 116–125. doi:10.1145/3144457.3144479.

[18] P. Kharat, V. Chaudhari, H. Maurya, S. Junghare, An inventory management using cloud function and protocol buffer for improved efficiency, IET Conference Proceedings 2023 (2023) 253–260. doi:10.1049/icp.2023.1499.

[19] F. Khodadadi, R. O. Sinnott, A Semantic-aware Framework for Service Definition and Discovery in the Internet of Things Using CoAP, Procedia Computer Science 113 (2017) 146–153. doi:10.1016/j.procs.2017.08.334.

[20] T. F. Ilyas, F. Arkan, R. Kurniawan, T. H. Budianto, G. B. Putra, Thingsboard-based prototype design for measuring depth and ph of kulong waters, IOP Conference Series: Earth and Environmental Science 926 (2021). doi:10.1088/1755-1315/926/1/012025.

[21] M. Henschke, X. Wei, X. Zhang, Data Visualization for Wireless Sensor Networks Using ThingsBoard, in: 2020 29th Wireless and Optical Communications Conference, WOCC 2020, 2020, pp. 1–6. doi:10.1109/WOCC48579.2020.9114929.

[22] L. O. Aghenta, M. T. Iqbal, Design and implementation of a low-cost, open source IoT-based SCADA system using ESP32 with OLED, ThingsBoard and MQTT protocol, AIMS Electronics and Electrical Engineering 4 (2019) 57–86. doi:10.3934/ElectrEng.2020.1.57.

[23] L. T. De Paolis, V. De Luca, R. Paiano, Sensor data collection and analytics with thingsboard and spark streaming, in: EESMS 2018 - Environmental, Energy, and Structural Monitoring Systems, Proceedings, 2018, p. 1 – 6. doi:10.1109/EESMS.2018.8405822.

[24] M. Casillo, F. Colace, M. De Santo, A. Lorusso, R. Mosca, D. Santaniello, VIOT_Lab: A Virtual Remote Laboratory for Internet of Things Based on ThingsBoard Platform, in: 2021 IEEE Frontiers

in Education Conference (FIE), 2021, pp. 1–6. doi:10.1109/FIE49875.2021.9637317.

[25] E. Okhovat, M. Bauer, Monitoring the Smart City Sensor Data Using Thingsboard and Node-Red, in: 2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI), 2021, pp. 425–432. doi:10.1109/SWC50871.2021.00064.

[26] D. N. Bestari, A. Wibowo, An IoT-Based Real-Time Weather Monitoring System Using Telegram Bot and Thingsboard Platform, International Journal of Interactive Mobile Technologies 17 (2023) 4–19. doi:10.3991/ijim.v17i06.34129.

[27] A. C. Sabuncu, K. A. Thornton, Leveraging ThingsBoard IoT Service for Remote Experimentation, in: ASEE Annual Conference and Exposition, Conference Proceedings, American Society for Engineering Education, 2022. URL: https://peer.asee.org/leveraging-thingsboard-iot-service-for-remote-experimentation.pdf.

[28] S. I. Jang, J. Y. Kim, A. Iskakov, M. Fatih Demirci, K. S. Wong, Y. J. Kim, M. H. Kim, Blockchain Based Authentication Method for ThingsBoard, in: J. J. Park, S. J. Fong, Y. Pan, Y. Sung (Eds.), Advances in Computer Science and Ubiquitous Computing, Springer, Singapore, 2021, pp. 471–479. doi:10.1007/978-981-15-9343-7_65.

[29] T. M. Kadarina, R. Priambodo, Monitoring heart rate and SpO2 using Thingsboard IoT platform for mother and child preventive healthcare, IOP Conference Series: Materials Science and Engineering 453 (2018) 012028. doi:10.1088/1757-899X/453/1/012028.

[30] JSON, JSON.org. Introducing JSON, 2001. URL: https://www.json.org/json-en.html.

[31] Google, Protocol Buffers, 2023. URL: https://developers.google.com/protocol-buffers.

[32] ThingsBoard, ThingsBoard IoT Platform, 2016. URL: https://thingsboard.io.

[33] Efento, Efento NB-IoT sensors and ThingsBoard, 2024. URL: https://getefento.com/library/efento-nb-iot-sensors-and-things-board/.

[34] ThingsBoard, Device Profiles, 2023. URL: https://thingsboard.io/docs/user-guide/device-profiles/.

Test platform for Simulation-In-Hardware of unmanned aerial vehicle on-board computer

Arsen R. Petrosian[1], Ruslan V. Petrosian[1] and Oleksandra M. Svintsytska[1]

[1]Zhytomyr Polytechnic State University, 103 Chudnivsyka Str., Zhytomyr, 10005, Ukraine

Abstract

Recent years have been marked by the rapid development of unmanned aerial vehicles, which is of interest not only to ordinary citizens but also to military, industrial and civilian spheres of activity. Designing and developing software to control the orientation and movement of an unmanned aerial vehicle in space takes a long time, including time for debugging, testing, and conducting flight tests. Errors made in the development of the software can lead to emergencies or other unforeseen failures during operation, which can lead to the destruction of the vehicle or cause harm to people and the environment. In the course of analyzing recent studies of software testing methods for embedded systems, their advantages and disadvantages were identified. It was found that both mechanical test benches and simulation modeling in a 3D environment are used: SIL, HIL and SIH simulation. Mechanical test benches allow testing and calibration of the parameters of real models of unmanned aerial vehicles, but have some limitations. Simulation modeling does not require additional equipment due to the use of virtual models of unmanned aircraft and does not have the limitations inherent in mechanical test benches. The most successful implementation of the test platform is the implementation using SIH simulation. The proposed method is an improved version of SIH simulation. The basis of this variant is a sensor and actuator simulator that ensures the operation of the original firmware of an unmanned aerial vehicle and allows you to organize the interconnection of on-board and personal computers. To test the idea, the test platform was implemented in practice. A configuration utility for the simulator of sensors and actuators has also been developed. The results obtained in the work will allow to organize software testing of the on-board computer of an unmanned aerial vehicle even without having the source code.

Keywords

UAV, unmanned aerial vehicle, on-board computer, software testing, SIH simulation, HIL simulation, 3D environment, sensor and actuator simulator

1. Introduction

Unmanned aerial vehicles (UAV), which are flying robots, are an important part of scientific research in military, industrial and civilian areas of activity: aerial photography and mapping, promptly obtaining information about the consequences of emergencies, monitoring industrial and natural complexes, delivering goods, entertainment purposes, etc. Designing and developing software for controlling UAV orientation in space takes a long time, including time for debugging, testing, and flight tests.

Using classical methods of debugging and testing flight controller firmware is problematic. One way to test controller firmware is to use an oscilloscope and logic analyzer to obtain timing diagrams. This approach is effective, but it is not always possible. In our case it is not possible to use these tools, because the UAV must move in space. When the controller is operating power equipment, it must not be stopped. Stopping the controller during operation at the breakpoint will, at best, cause the equipment to stop, and at worst it may lead to equipment failure. Also further step-by-step debugging becomes impossible because the system state will change (closed-loop control system). However, many debugging tools, such as J-Link, display the change of variables in real time. As in the first case, it is impossible to connect directly, so it is necessary to use remote data transfer. this possibility is theoretically possible (figure 1), but this option is not provided.

Each of the considered options can be used to solve narrowly focused problems, but all the considered approaches do not guarantee the work of the whole system. Mistakes made in the development of UAV

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine

xckaytz@gmail.com (A. R. Petrosian); e_rvs@ukr.net (R. V. Petrosian); sasha_1904@ukr.net (O. M. Svintsytska)
0000-0003-0960-8461 (A. R. Petrosian); 0000-0002-0388-8821 (R. V. Petrosian); 0000-0002-2613-2437 (O. M. Svintsytska)

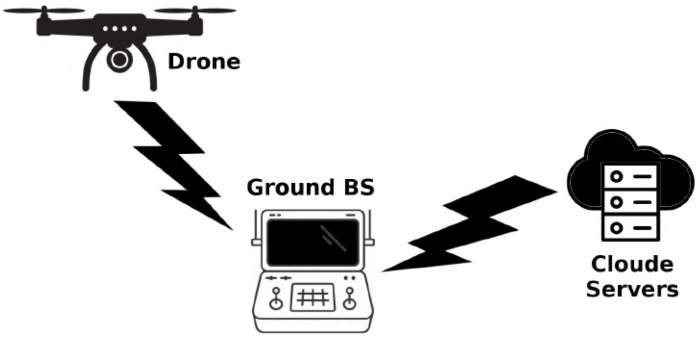


Figure 1: Simplified structure of the UAV peripheral computing network.

software can lead to emergencies or other unforeseen failures during operation. In this regard, the task of checking the reliability of the UAV's on-board computer software arises, which will simplify the maintenance, modernization and optimization of the software.

2. Theoretical background

Debugging and testing UAV software is a challenging task. The problem lies in the difficulty of generating a complete set of sensor signals and the impossibility of creating emergency situations on a real UAV, so special software and hardware are used to develop, test and debug UAV software.

Let us consider modern approaches used to verify the reliability of software during development, namely testing. According to the degree of code isolation, there are 4 levels of testing [1]:

- unit testing;
- integration testing;
- system testing;
- acceptance testing.

Unit testing is a level in which individual modules or components of a program (functions, methods, or classes) are tested independently of the rest of the program. This level of testing is intended to verify that each component works correctly. An important aspect of unit testing is the isolation of the component under test from other components of the application to ensure that bugs are detected and corrected locally within the component.

Integration testing is a level that is aimed at checking the interaction between different modules or components of the application. Integration testing checks how components interact with each other and how they interact with external systems, if any.

System testing – a level that is carried out at the final stage of development, when all components of the application are already integrated together and ready for testing in real conditions before the application is released into operation

Acceptance testing is a level aimed at verifying that the application meets the requirements of the customer or end user. It is usually performed by the customer or their representatives.

Testing ensures that:

- code quality;
- compliance with the stated requirements;
- optimization of computing resources;
- data security and integrity;
- time saving (searching for errors can take a lot of time);

- etc.

In many cases, the use of testing during the development of, for example, computer application software is sufficient to ensure the required level of reliability of the software, but not for firmware.

For effective and efficient testing of embedded software, a large number of testing methods, approaches, and tools are used, so in [2], a review and systematization of literature sources in this area was carried out.

Embedded systems often need to work in interaction with the environment, using sensors to obtain input data (temperature, pressure, etc.) [3], and UAVs are no exception. In [4], the main idea is to develop a test platform for mechanical flight simulation and better stabilization of multi-rotor UAVs using various feedback control algorithms, including PID controllers [5, 6]. This stand allows you to check and calibrate model parameters and perform real-time control of a multi-rotor UAV. A more functional stand for educational purposes is presented in [7]. As in the previous work, the test stand is a gyroscopic structure with three degrees of freedom, which provides pitch, roll, and yaw motion of the quadrotor. However, unlike the previous test stand, it allows translational movement, albeit with limitations (figure 2).

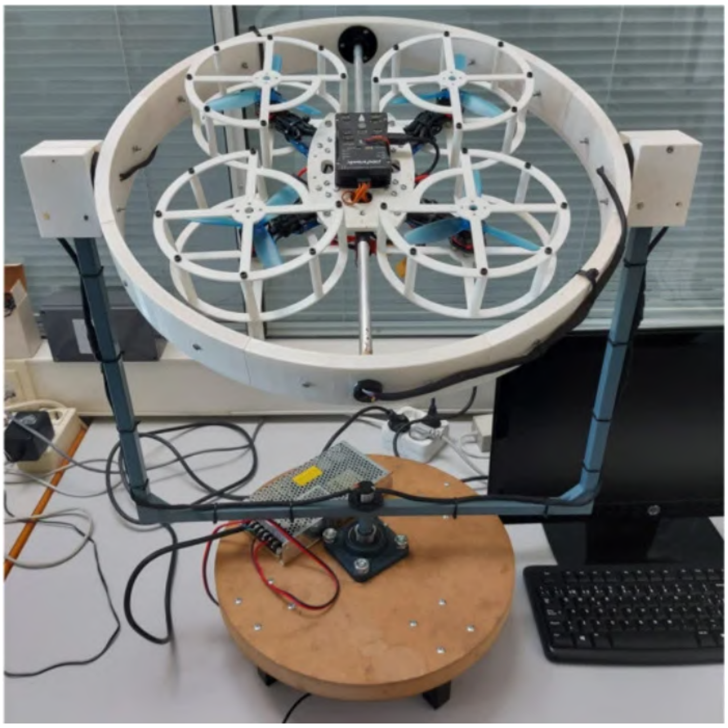


Figure 2: Experimental prototype with 3 degrees of freedom for testing control algorithms in a quadcopter.

One of the effective methods of testing embedded systems is simulation modeling. Simulation modeling is a research method that uses models that describe real-world processes. Using such models, it is possible to test the real system without exposing it to danger. Simulation modeling is most often used when it is impossible to conduct an experiment on a real object or it is necessary to simulate the behavior of a real system in time.

The firmware test in UAV simulation can be performed in three places: on a host computer, using code compilation for the source platform; in an emulator on a host computer, such as QEMU; on a real flight controller, using cross-compilation of code for the target hardware. The first two options are used when performing SIL (Software in the Loop) simulation [8, 9]. The advantage of SIL is that it is easy to organise, as no additional hardware is required. SIL allows developers to perform firmware simulation in the early stages of development, even before it is integrated into the target hardware. The third option is used when performing HIL (Hardware in the Loop) simulation [8, 10, 11]. HIL simulation involves the use of target equipment that brings the system's operation as close as possible to real conditions. Recently, another approach to testing has emerged – SIH simulation [12]. SIH simulation is an alternative to HIL simulation. In this case, everything works on the onboard computer, and the PC is used only to display the virtual UAV.

3. Result

As mentioned above, various testing methods and tools are used to ensure the reliability of embedded system software. The main difficulty is testing the hardware components of the embedded system, so additional hardware and/or software tools are being developed. Figure 2 shows a test stand that provides verification of the functioning of algorithms in the on-board computer, the operation of sensors, motors, etc. However, it is obvious that this stand limits the UAV's movement in space. Another approach that allows you to test a full-fledged UAV flight is to use HIL simulation. The flight is performed in a 3D environment on a computer (e.g., Gazebo, jMavSim, or another). Figure 3 shows a diagram of such a HIL simulation system of the PX4 autopilot software [11].

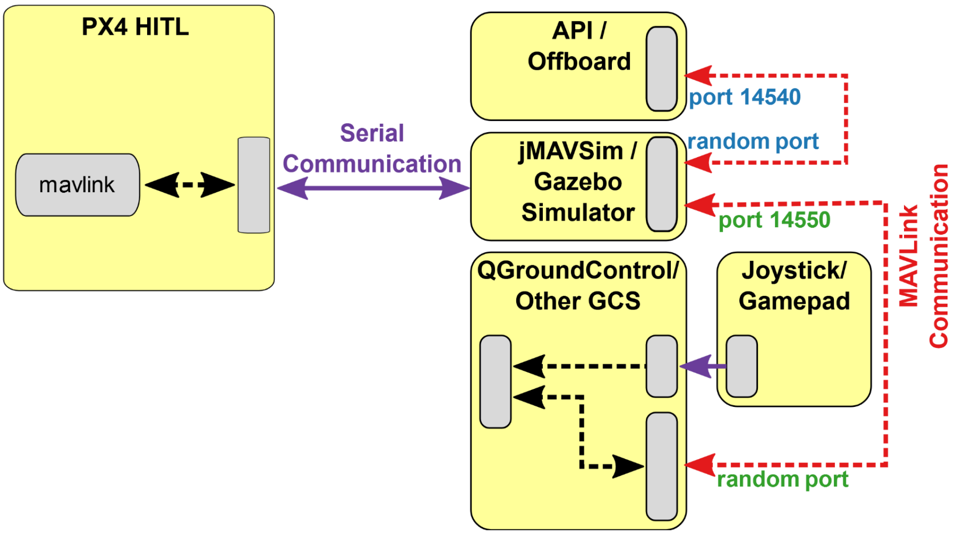


Figure 3: Block diagram of HIL simulation in PX4.

Obviously, this approach allows testing only the operation of the on-board computer, while other hardware nodes are virtual. These virtual nodes interact with the on-board computer using the MAVLink protocol [13]. Figure 4 shows the appearance of the jMavSim 3D environment.

To ensure firmware testing in the on-board computer, you need to use mock objects. The purpose of mock objects is to replace the objects that are tested in the program code with equivalent debugging objects. Creating mock objects can be associated with some difficulties, for example, using interrupts from a specific communication interface that a real sensor uses. The disadvantage of this approach is that the firmware during testing will not correspond to its final implementation. The use of SIH simulation (figure 5) improves the situation when testing UAVs [12], but does not eliminate most of the problems that HIL simulation has.

To address these shortcomings, it is proposed to implement a test platform that uses the original firmware of the on-board computer. How to implement it? Consider a simplified diagram of any sensor (figure 6).

The sensor consists of:

- a sensing element that is directly related to the measured value;
- transducer, which serves to convert the measured value into another value convenient for transmission, processing and storage.

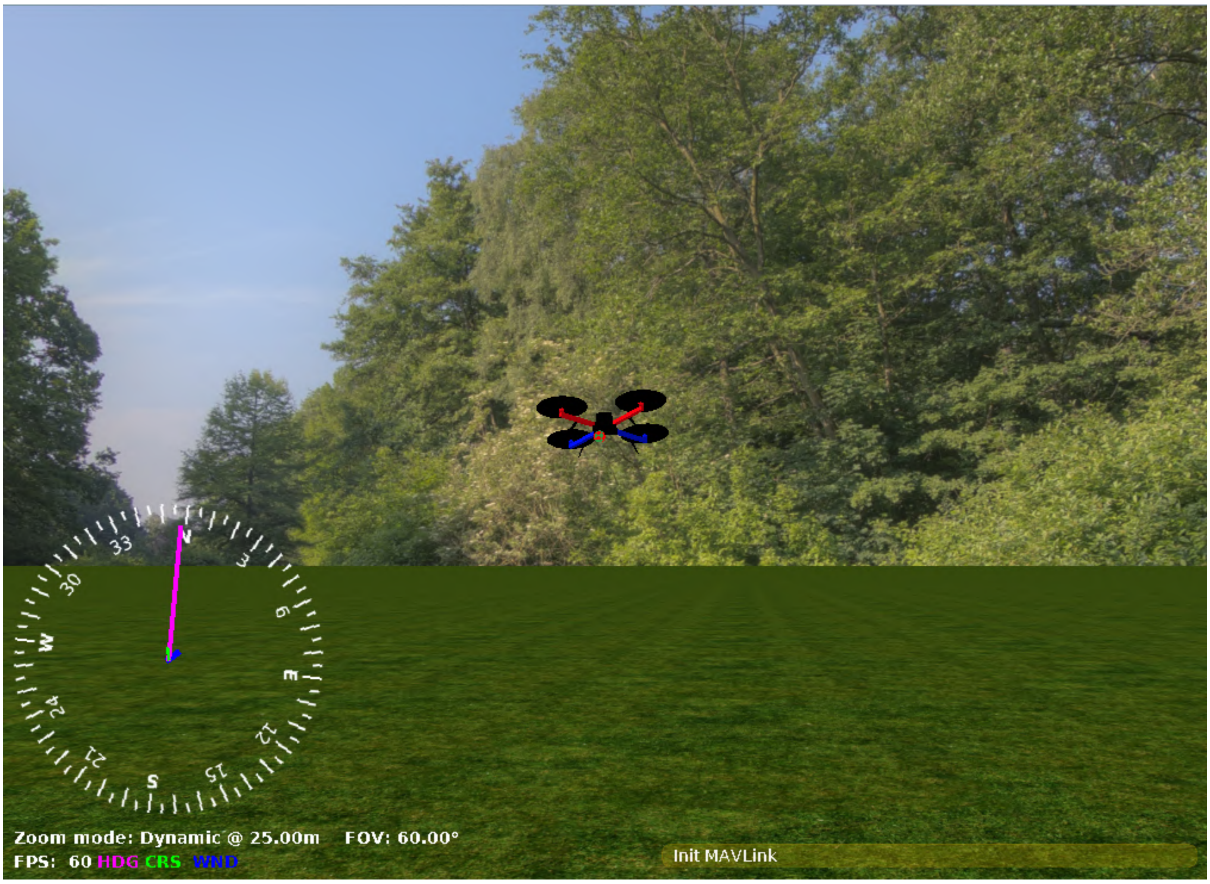


Figure 4: jMavSim 3D environment window.

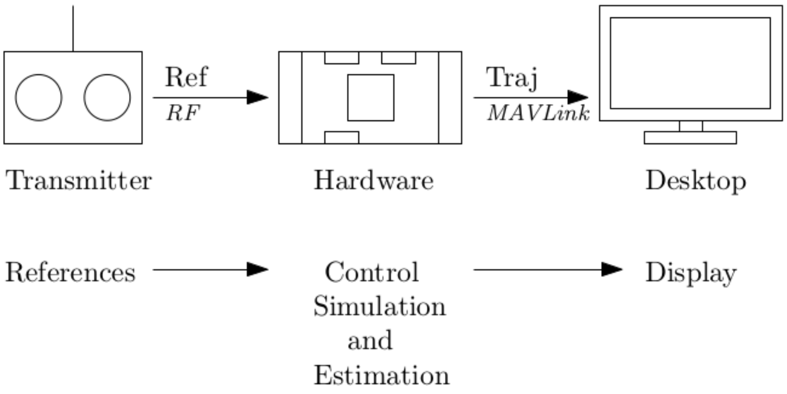


Figure 5: Block diagram of SIH simulation in PX4.

When using any simulation option (HIL, SIH), the virtual sensor is fully implemented in a 3D environment [14, 15]. In this case, the question arises: how to transmit information to the on-board computer? The MAVLink protocol is mainly used, although it is not necessary, for which mock objects are implemented in the firmware of the on-board computer. Thus, the data of the virtual sensor is processed on the side of the on-board computer. Obviously, in this case, it is necessary to complicate the architecture of the autopilot software, as well as to provide for different compilation scenarios. However, the KISS design principle states that there is no need to overcomplicate, each node should perform only a specific task.

The on-board computer already uses communication interfaces with sensors (mainly I2C, SPI), so it is

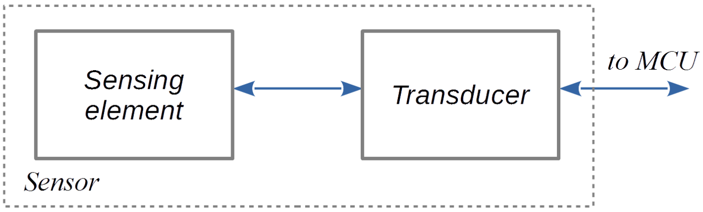


Figure 6: Simplified block diagram of a sensor.

logical to use them to solve the problem. In this case, the on-board computer will access real hardware nodes, but this is problematic because these interfaces are not available in a personal computer. On the other hand, it is necessary to use a 3D environment to organise safe flights when testing UAV algorithms.

Obviously, a node is needed to connect the on-board computer with the 3D environment on a PC to transmit navigation and control information to the UAV. From the above, it follows that a sensor and actuator simulator (SAS) is needed, where the software model will correspond to the hardware nodes. The architecture of the proposed test platform is shown in figure 7.

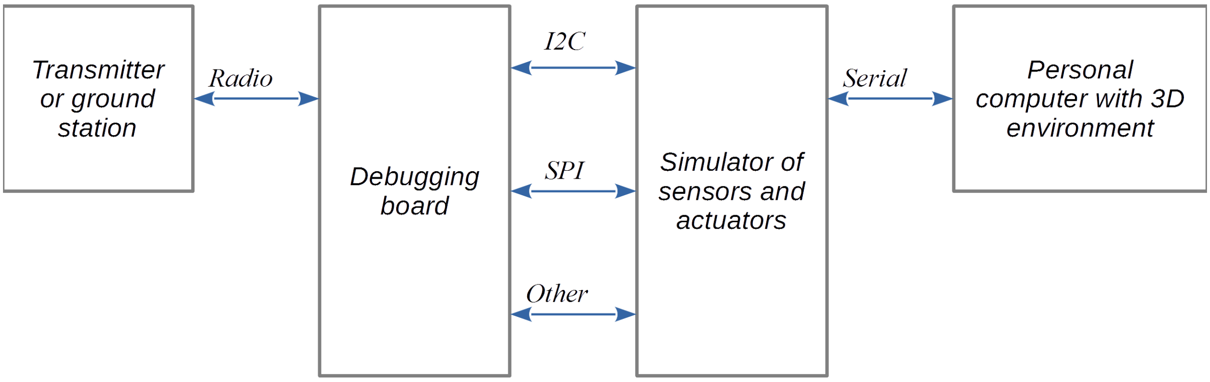

Transmitter or ground station
Radio
Debugging board
I2C
SPI
Other
Simulator of sensors and actuators
Serial
Personal computer with 3D environment

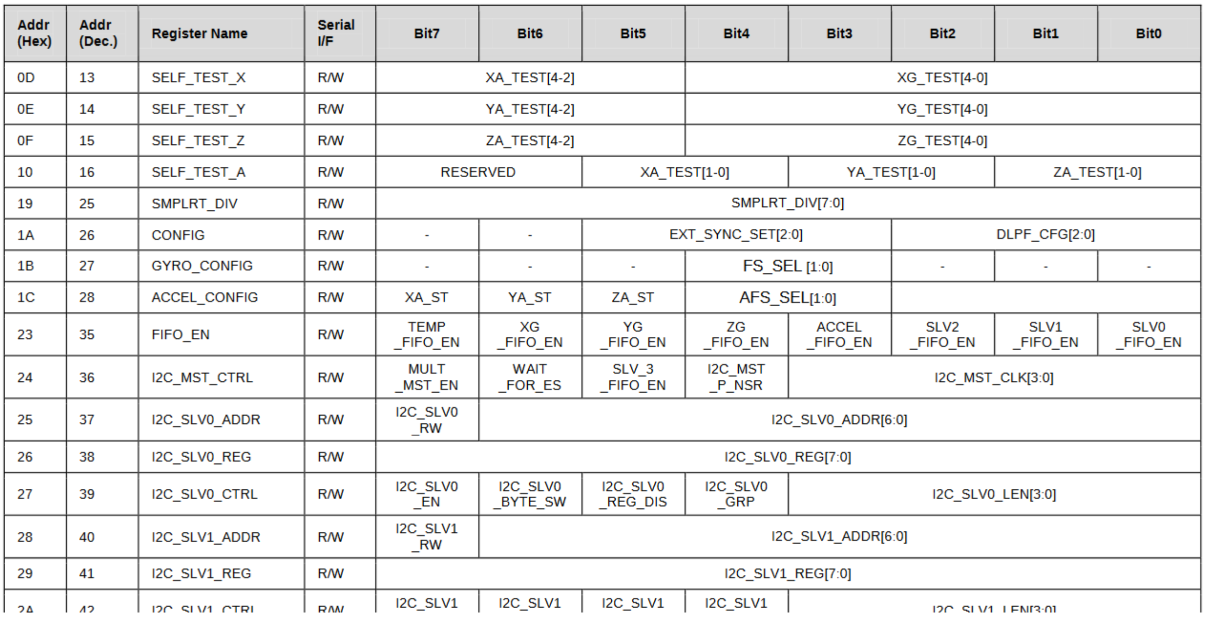
Figure 7: Architecture of the test platform.

When using this approach, in fact, the SAS will perform the functions of a sensor converter (figure 6), and only the sensing element will be used in the 3D environment. The information transmitted from the 3D environment will be wrapped in a software model of the sensor (in our case, MPU-6050 [16, 17]). Figure 8 shows some registers of the MPU-6050 accelerometer and gyroscope.

A debugging board is required for UAV simulation. The most popular microcontrollers for the on-board computer are STM32F405RGT6 and STM32F722RET6. Unfortunately, STMicroelectronics does not produce a debugging board based on the STM32F405RGT6 microcontroller, so you need to use a third-party debugging board, such as the Core405R from Waveshare or the STM32F405 Core Board from WeAct. The NUCLEO-F722ZE debug board can be used to simulate an on-board computer based on the STM32F722RET6 microcontroller. The SAS can be performed on any debugging board that has three I2C and SPI interfaces each (the number of interfaces is determined by the number of interfaces in the above-mentioned microcontrollers). In our case, we used the STM32G431 Core Board from WeAct. The configuration of the SAS for each sensor is performed similarly to its prototype, and the data registers are filled with information coming from a personal computer with a 3D environment.

The software of the SAS is implemented in the C++ programming language. For sensors with an I2C interface, the basis is an abstract class shown in the screen (figure 9).

To configure the SAS, we developed a simple utility in the C++ programming language using the QT framework (figure 10). For each channel, the sensor model, interface, and variable part of the address,



Addr (Hex)	Addr (Dec.)	Register Name	Serial I/F	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0D	13	SELF_TEST_X	R/W	XA_TEST[4-2]			XG_TEST[4-0]				
0E	14	SELF_TEST_Y	R/W	YA_TEST[4-2]			YG_TEST[4-0]				
0F	15	SELF_TEST_Z	R/W	ZA_TEST[4-2]			ZG_TEST[4-0]				
10	16	SELF_TEST_A	R/W	RESERVED		XA_TEST[1-0]		YA_TEST[1-0]		ZA_TEST[1-0]	
19	25	SMPLRT_DIV	R/W	SMPLRT_DIV[7:0]							
1A	26	CONFIG	R/W	-	-	EXT_SYNC_SET[2:0]			DLPF_CFG[2:0]		
1B	27	GYRO_CONFIG	R/W	-	-	-	FS_SEL [1:0]		-	-	-
1C	28	ACCEL_CONFIG	R/W	XA_ST	YA_ST	ZA_ST	AFS_SEL[1:0]				
23	35	FIFO_EN	R/W	TEMP _FIFO_EN	XG _FIFO_EN	YG _FIFO_EN	ZG _FIFO_EN	ACCEL _FIFO_EN	SLV2 _FIFO_EN	SLV1 _FIFO_EN	SLV0 _FIFO_EN
24	36	I2C_MST_CTRL	R/W	MULT _MST_EN	WAIT _FOR_ES	SLV_3 _FIFO_EN	I2C_MST _P_NSR	I2C_MST_CLK[3:0]			
25	37	I2C_SLV0_ADDR	R/W	I2C_SLV0 _RW	I2C_SLV0_ADDR[6:0]						
26	38	I2C_SLV0_REG	R/W	I2C_SLV0_REG[7:0]							
27	39	I2C_SLV0_CTRL	R/W	I2C_SLV0 _EN	I2C_SLV0 _BYTE_SW	I2C_SLV0 _REG_DIS	I2C_SLV0 _GRP	I2C_SLV0_LEN[3:0]			
28	40	I2C_SLV1_ADDR	R/W	I2C_SLV1 _RW	I2C_SLV1_ADDR[6:0]						
29	41	I2C_SLV1_REG	R/W	I2C_SLV1_REG[7:0]							
2A	42	I2C_SLV1_CTRL	R/W	I2C_SLV1	I2C_SLV1	I2C_SLV1	I2C_SLV1	I2C_SLV1_LEN[3:0]			

Figure 8: MPU-6050 register map.

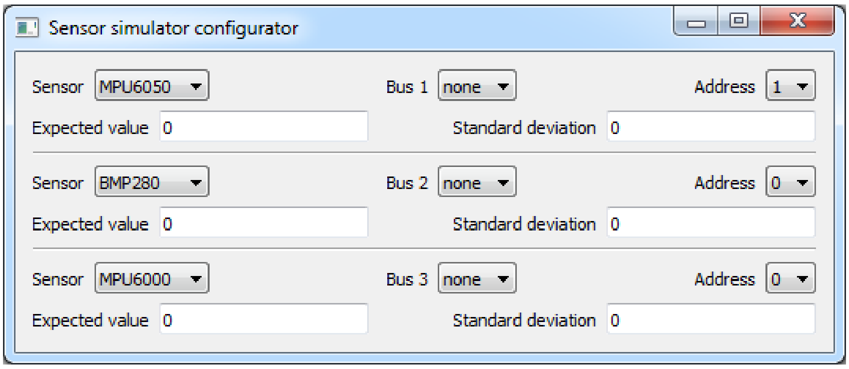
```
class I2C {
private:
    // Sensor address on the I2C bus
    uint32_t i2c_address;
public:
    // Constructor
    I2C(uint32_t address) : i2c_address(address) {}
    // Destructor
    virtual ~I2C() {}
    // Function returns the address of the sensor on the I2C bus
    uint32_t get_i2c_address() {return i2c_address;}
    // Function sets the address of the sensor on the I2C bus
    void set_i2c_address(uint32_t address) {i2c_address=address;}
    // Function is called when accessing the sensor via I2C bus
    virtual bool i2c_connect() = 0;
    // Function is called when reading data bytes from the sensor
    virtual void i2c_disconnect() = 0;
    // Function is called when a data byte is written to the sensor
    virtual uint8_t i2c_read() = 0;
    // Function is called when communication with the sensor is completed
    virtual bool i2c_write(uint8_t data) = 0;
};
```

Figure 9: Abstract class of sensors with I2C interface.

as well as noise parameters are configured in accordance with expression (1) [14]:

$$y_m = y + b + w_y, \tag{1}$$

where y is the original value, b is the current offset, and w_y is a white Gaussian noise with zero mean.



Sensor simulator configurator

Sensor MPU6050 Bus 1 none Address 1
Expected value 0 Standard deviation 0

Sensor BMP280 Bus 2 none Address 0
Expected value 0 Standard deviation 0

Sensor MPU6000 Bus 3 none Address 0
Expected value 0 Standard deviation 0

Figure 10: Appearance of the configurator of the sensor and actuator simulator.

4. Conclusion

The article considers a test platform for hardware modelling of an unmanned aircraft's onboard computer. Various testing methods and tools are used to ensure the reliability of embedded system software. The main difficulty is testing the hardware components of the embedded system, so additional hardware and software tools are being developed.

To accomplish this task, a comparative analysis of existing testing approaches was carried out to identify advantages and disadvantages. It was found that both mechanical test stand and simulation modelling in a 3D environment are used: SIL and HIL. A mechanical test rig is heavy and restricts the UAV's movement in space. Existing SIL and HIL simulators do not require additional equipment, but they do not take into account the actual operation of hardware components.

An improved version of SIH simulation is proposed. The proposed variant is based on a sensor and actuator simulator that ensures the operation of the original UAV firmware and allows for the interconnection of the onboard and personal computer. The sensor and actuator simulator was built on the basis of the Core Board STM32G431 debugging board from WeAct. To configure the sensor and actuator simulator, a small utility was developed in C++ using the QT framework.

Acknowledgments

Conceptualization, Arsen Petrosian and Ruslan Petrosian; software, Arsen Petrosian; hardware, Ruslan Petrosian; investigation, Arsen Petrosian; resources, Oleksandra Svintsytska; data curation, Oleksandra Svintsytska; writing-review and editing, Arsen Petrosian.

We would like to express our gratitude to our relatives who supported and helped us. We would also like to express our gratitude to all the organizers of the conference “doors-2024: 4rd Edge Computing Workshop” conference, especially Tetiana Vakaliuk.

References

[1] M. Aniche, Effective Software Testing: A developer's guide, Simon and Schuster, 2022.

[2] V. Garousi, M. Felderer, Ç. M. Karapıçak, U. Yılmaz, Testing embedded software: A survey of the literature, Information and Software Technology 104 (2018) 14–45. doi:10.1016/j.infsof.2018.06.016.

[3] A. Banerjee, S. Chattopadhyay, A. Roychoudhury, On testing embedded software, in: Advances in Computers, volume 101, Elsevier, 2016, pp. 121–153. doi:10.1016/bs.adcom.2015.11.005.

[4] M. Hancer, R. Bitirgen, I. Bayezit, Designing 3-DOF hardware-in-the-loop test platform controlling multirotor vehicles, IFAC-PapersOnLine 51 (2018) 119–124. doi:10.1016/j.ifacol.2018.06.058.

[5] R. V. Petrosian, I. A. Pilkevych, A. R. Petrosian, Algorithm for optimizing a PID controller model based on a digital filter using a genetic algorithm, CEUR Workshop Proceedings 3374 (2023) 97–111. URL: https://ceur-ws.org/Vol-3374/paper07.pdf.

[6] M. Zhang, C. Xu, D. Xu, G. Ma, H. Han, X. Zong, Research on improved sparrow search algorithm for PID controller parameter optimization, Bulletin of the Polish Academy of Sciences Technical Sciences 71 (2023) e147344–e147344. doi:10.24425/bpasts.2023.147344.

[7] U. Veyna, S. Garcia-Nieto, R. Simarro, J. V. Salcedo, Quadcopters Testing Platform for Educational Environments, Sensors 21 (2021) 4134. doi:10.3390/s21124134.

[8] C. Coopmans, M. Podhradský, N. V. Hoffer, Software- and hardware-in-the-loop verification of flight dynamics model and flight control simulation of a fixed-wing unmanned aerial vehicle, in: 2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS), 2015, pp. 115–122. doi:10.1109/RED-UAS.2015.7440998.

[9] K. D. Nguyen, C. Ha, J. T. Jang, Development of a New Hybrid Drone and Software-in-the-Loop Simulation Using PX4 Code, in: D.-S. Huang, V. Bevilacqua, P. Premaratne, P. Gupta (Eds.), Intelligent Computing Theories and Application, Springer International Publishing, Cham, 2018, pp. 84–93. doi:10.1007/978-3-319-95930-6_9.

[10] K. D. Nguyen, C. Ha, Development of Hardware-in-the-Loop Simulation Based on Gazebo and Pixhawk for Unmanned Aerial Vehicles, International Journal of Aeronautical and Space Sciences 19 (2018) 238–249. doi:10.1007/s42405-018-0012-8.

[11] Hardware in the Loop Simulation. PX4 Autopilot User Guide, 2023. URL: https://docs.px4.io/main/en/simulation/hitl.html.

[12] Simulation-In-Hardware. PX4 Autopilot User Guide, 2023. URL: https://dev.px4.io/v1.9.0_noredirect/en/simulation/simulation-in-hardware.html.

[13] Protocol Overview. MAVLink Developer Guide, 2023. URL: https://mavlink.io/en/about/overview.html.

[14] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, O. von Stryk, Comprehensive Simulation of Quadrotor UAVs Using ROS and Gazebo, in: I. Noda, N. Ando, D. Brugali, J. J. Kuffner (Eds.), Simulation, Modeling, and Programming for Autonomous Robots, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 400–411. doi:10.1007/978-3-642-34327-8_36.

[15] A. I. Hentati, L. Krichen, M. Fourati, L. C. Fourati, Simulation Tools, Environments and Frameworks for UAV Systems Performance Analysis, in: 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC), 2018, pp. 1495–1500. doi:10.1109/IWCMC.2018.8450505.

[16] TDK, MPU-6000 and MPU-6050. Product Specification, 2013. URL: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf.

[17] TDK, MPU-6000 and MPU-6050. Register Map and Descriptions, 2013. URL: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Register-Map1.pdf.

Data security of IoT devices with limited resources: challenges and potential solutions

Inna Rozlomii[1,2], Andrii Yarmilko[1] and Serhii Naumenko[1]

[1]*Bohdan Khmelnytsky National University of Cherkasy, 81 Shevchenko Blvd., Cherkasy, 18031, Ukraine*
[2]*Cherkasy State Technological University, 460 Shevchenko Blvd., Cherkasy, 18006, Ukraine*

Abstract
Integration of the Internet of Things (IoT) into various application domains not only expands capabilities but also brings forth a multitude of challenges. These challenges revolve around the security of IoT devices, many of which are characterized by limited resources such as memory, power consumption, and computational power. This article examines key challenges associated with ensuring the security of IoT devices and proposes potential solutions and strategies adapted to resource constraints. Emphasis is placed on the development and analysis of lightweight cryptographic algorithms capable of providing robust data protection with minimal resource utilization. Strategies for efficient energy management and memory usage optimization are also discussed, critical for ensuring the stable and uninterrupted operation of IoT devices. The article highlights the necessity of developing adaptive security mechanisms that can effectively respond to dynamic operational conditions and resource constraints. The key importance of continuously updating security mechanisms to adapt to changing conditions and to guard against new and future cyber threats is underscored. In addition to technical aspects, the importance of strategic planning and innovation in IoT security is also illuminated. It is noted that further research and development should focus on creating integrated solutions that combine hardware, software, and managerial aspects to optimize overall efficiency and security of IoT systems. This article contributes to the understanding and resolution of security issues in IoT devices operating under resource constraints. It provides a broad overview of existing challenges and opportunities while suggesting directions for future research and development in this dynamically evolving field.

Keywords
IoT, limited resources, cryptographic algorithms, energy efficiency, memory management, authentication algorithms, cyber threats

1. Introduction

Embedded Internet of Things (IoT) devices are compact, integrated devices embedded in various objects capable of collecting, processing, and utilizing data, as well as exchanging it over a network without direct human involvement [1]. These devices provide automation and monitoring in various fields, from household systems to industrial processes, using their own computational resources to perform their functions [2, 3].

The significance of embedded IoT devices lies in their ability to add intelligence and functionality to different systems, facilitating data collection, process automation, and productivity enhancement [4]. They have become an essential element in advancing technologies and the development of the connected world [5].

However, the security of embedded IoT devices has become a key issue limiting their application [6]. This problem arises from the imbalance between the potentially high functionality of such devices and their resource constraints [7]. The limited computational power and memory resources of embedded devices typically complicate the implementation of robust security mechanisms to prevent unauthorized access to data and control the flow of information processes [8]. This poses serious challenges in ensuring security, as these devices may become targets of external attacks with critical consequences of both technical and humanitarian-legal nature [9, 10].

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine
inna-roz@ukr.net (I. Rozlomii); a-ja@ukr.net (A. Yarmilko); naumenko.serhii1122@vu.cdu.edu.ua (S. Naumenko)
0000-0001-5065-9004 (I. Rozlomii); 0000-0003-2062-2694 (A. Yarmilko); 0000-0002-6337-1605 (S. Naumenko)

Overcoming challenges related to the security of embedded IoT devices becomes a critical task in the context of their widespread integration into our everyday living spaces and industrial environments. The limited resources of these devices pose not only technical challenges but also serious potential consequences for user safety and infrastructure security. Failure to address the issue of limited resources and inadequate protection may lead to uncontrolled widespread access to confidential information, destruction of critical systems, or even the use of devices for malicious purposes.

The main security challenges of embedded IoT devices are associated with their inadequate protection and vulnerability to cyber attacks due to limited support for encryption and authentication, as well as insufficient capabilities for detecting and responding to potential threats. The aim of this research is to develop effective cryptographic protection strategies for embedded IoT devices with limited resources.

In the context of researching the security of embedded IoT devices, it is important to identify development perspectives aimed at ensuring their security and reliability. The development of effective cryptographic protection strategies is a key element of this process. Applying modern encryption and authentication methods will improve the reliability and accountability of embedded devices, providing a high level of protection in conditions of limited resources.

2. Related works

There are numerous studies dedicated to the security issues of embedded IoT devices with limited resources [11, 12]. Many of them indicate that the physical constraints of such devices complicate the implementation of comprehensive security measures and are the primary cause of numerous vulnerabilities [13]. In these security studies, the importance of embedded IoT device security has garnered significant interest due to their crucial role in daily life, industry, and infrastructure. Many works highlight the fundamental challenge of mismatch between data protection needs and the limited resources of the devices [14].

Security of embedded IoT devices has been the subject of many works investigating vulnerability issues related to limited computational capabilities, restricted memory capacity, and limited power supply [15, 16]. These studies have demonstrated that resource constraints impact the effectiveness of cryptographic protection and authentication processes, making devices vulnerable to external threats.

A significant research theme has been cryptographic protection strategies with limited resources [17]. Previous research has shown the low efficiency of certain cryptographic methods and proposed the use of lightweight encryption and authentication methods that require fewer resources [18, 19]. However, some studies suggest that lightweight methods may have their own limitations and require a balance between efficiency and security [20]. Awareness of these aspects is crucial for developing optimal cryptographic protection strategies for embedded IoT devices with limited resources.

Further research should focus on effective methods to ensure the security of embedded IoT devices, considering resource constraints and the requirement for a high level of protection. Emphasizing efficient authentication and cryptographic mechanisms that take into account limited resources is identified as a key direction for future scientific research in this area.

3. Capabilities of embedded devices and their resource limitations

The architecture of embedded IoT devices is presented as the interaction of three main components [21]:

1. Sensors and actuators – components that provide data collection and transmission. Sensors gather information from the environment (temperature, humidity, etc.), while actuators perform corresponding actions (e.g., turning devices on/off).
2. Data processor is responsible for processing and analyzing the collected data. This can be a microcontroller or a specialized computing system.
3. Network interface facilitates communication with the external environment through various network protocols such as Wi-Fi, Bluetooth, LoRa, Zigbee, etc.

The primary functions of embedded IoT devices include [22]:

1. Data collection – obtaining information from sensors.
2. Data processing – analyzing and processing the acquired data to perform defined tasks.
3. Actuator control – sending signals to actuators to execute specific actions.

The typical properties of embedded IoT devices, which form the core spectrum of their functional and technical advantages, are accompanied by limitations related to deployment platforms and methods of ensuring autonomy. In general, these limitations encompass the following aspects [23]:

1. Computational power: Embedded IoT devices have limited capability for complex computations due to restricted computational power. This may lead to constraints in applying advanced encryption algorithms and performing computationally intensive operations, reducing the device's security level.
2. Memory: The limited memory capacity in embedded devices complicates the storage of a large amount of data and software. This may result in a reduction of available resources for storing encryption keys, user data, and other critical elements, increasing vulnerability to attacks.
3. Power supply: mbedded IoT devices are often powered by autonomous energy sources or have limited power consumption. This limitation in power supply can lead to unforeseen interruptions in device operation or limit security capabilities, as the device may power off or enter a low-power mode, reducing its ability to detect and respond to potential threats.

These outlined limitations impact the capabilities of embedded devices in implementing robust security measures and pose a challenge in ensuring data reliability and protection.

4. Vulnerabilities of the security systems in embedded IoT devices

One of the key issues in the field of embedded IoT devices is the presence of vulnerabilities that can be exploited for attacks and security breaches. Securing embedded IoT devices becomes a crucial task as these devices are used in various life domains, ranging from household systems to critical infrastructure [15, 24, 25]. However, they also become a heightened focus for cybercriminals due to a range of vulnerabilities:

1. Inadequate Authentication and Authorization. A low level of authentication can serve as a starting point for unauthorized access to the device. The absence of robust user identity verification methods, the use of weak passwords, or simple authorization methods can be entry points for cyber-attacks. This can occur due to inadequate determination of access rights to device functions or data. In the absence of authentication, the likelihood of a successful attack on the device can be described by the following formula:

   $$P(A) = \frac{N_s}{N_t} \times 100\%, \tag{1}$$

   where $P(A)$ is the probability of an attack, N_s is the number of successful attacks, N_t is the total number of attack attempts.
2. Insufficient Cryptographic Protection: The use of weak or outdated encryption algorithms in IoT devices makes data more vulnerable to interception and compromise. If encryption employs keys of insufficient length or is vulnerable to known attacks, there is a risk of compromising the confidentiality and integrity of data, as well as threats to their availability. To determine the effectiveness of encryption, the Shannon encryption model can be utilized:

   $$C = log_2(1 + \frac{S}{N}), \tag{2}$$

   where C – the channel capacity, S – the signal power, N – the noise level.

3. Insufficient Software Updates: Limited memory in embedded devices can complicate the software update process. This creates a risk of temporary or permanent vulnerability of the device to new threats or vulnerabilities, as it may remain without updates to apply security patches or fix software defects that ensure security.

5. Security risks of embedded IoT devices

In the network of embedded IoT devices, ensuring security remains one of the main challenges. This is particularly crucial due to the limited resources characterizing these devices. Examining memory, energy consumption, and computational power issues, it can be observed that these aspects serve as potential security threats.

The limitation of memory in embedded systems complicates not only data storage but also the implementation of effective encryption methods. The reduced operational duration due to limited energy consumption becomes a starting point for potential DoS attacks. Additionally, limited computational power complicates the application of robust encryption and authentication methods.

Examining memory, energy consumption, and computational power, we can determine that:

- **Memory limitations** in embedded devices can lead to buffer overflows and constraints in storing encryption keys, complicating the cryptographic protection of information.
- **Energy supply** is a fundamental factor limiting the operational duration of devices and the risk of potential DoS attacks due to targeted expenditure of limited energy.
- **Limited computational power** complicates the application of complex encryption algorithms and may contribute to the execution of malicious code in case of insufficient input data validation.

The discussed limitations expose risks that need to be carefully considered and adequately addressed in embedded IoT devices to ensure the reliability, confidentiality, and integrity of the processed data.

5.1. Risks due to limited power consumption

Limited memory capabilities can cause issues in implementing cryptographic protection for embedded devices due to buffer overflows and restricted capacity for key storage:

1. Buffer overflow creates the possibility of embedding malicious code or executing code in vulnerable areas. The result is the emergence of vulnerabilities that can be exploited by attackers. Attacks leveraging these vulnerabilities may lead to system compromise, unauthorized code execution, or leakage of sensitive data.
2. As a result of the limited memory capabilities of embedded IoT devices to store encryption keys, there is a risk of their compromise. This is due to the complexity in the processes of storing and managing encryption keys, which are critical elements for ensuring data security. Typically, for system security, it is important to have diverse keys for various encryption tasks. However, due to limited memory, it may be challenging to provide the necessary volume of unique keys for data encryption.
3. Key management also becomes a challenge due to limited resources. For information security, keys need to be efficiently stored, updated, and rotated. However, limited memory can restrict the capacity for storing and processing key information, complicating their effective management. Thus, the complex storage and management of keys can serve as a foundation for their compromise. If keys are not stored or managed properly, it can make them more accessible to attackers or increase the likelihood of system vulnerabilities to attacks aimed at obtaining these keys.

Considering the limited memory capabilities of embedded IoT devices, cryptographic protection may become vulnerable due to buffer overflows and difficulties in storing encryption keys.

5.2. Risks arising from memory limitations

Energy consumption of an embedded system may be insufficient for the operation of cryptographic protection, both due to the design features of autonomous IoT module and intentional unauthorized impact on their power components. Threats related to energy consumption pose a wide range of security risks for the system:

1. Energy Attacks. Attacks aimed at reducing the energy consumption of IoT devices pose a serious threat to their normal functioning. These attacks can be implemented by constantly activating devices, prompting them to consume excessive energy. The consequence of such excessive energy consumption can be the depletion of the device's battery, leading to its shutdown or disruption of normal operation. This can be problematic, especially for devices operating on batteries or in conditions of limited power supply. Continuous excessive energy consumption can lead to a decrease in device performance and efficiency, making it more vulnerable to various types of attacks or limiting security capabilities due to insufficient energy for the normal operation of protective mechanisms.
2. Interruptions in Operation. Limited charge in an autonomous energy source can cause unforeseen interruptions in the device's operation, creating serious security risks. When energy becomes limited, the device may abruptly shut down or transition into a low-power mode. Such interruptions in operation can lead to a decrease in the device's reliability and may be exploited by malicious actors for attacks. As a result, data being processed or stored in the device at that moment may be lost or damaged. These unforeseen halts can create a window of opportunity for attacks on the device or its data, as they may be unavailable for protection or remain unprotected during such times.
3. Reaction Delays. Limited energy consumption in embedded IoT devices, aimed at energy conservation, can significantly impact their response time when detecting threats or attacks. This can lead to delays in identifying anomalies or responding to potential threats in the network. For energy-saving purposes, a device may operate in a standby mode, during which it is inactive or does not perform specific operations. In this mode, it may be less responsive to changes or anomalous situations, as it consumes a minimal amount of energy, affecting its ability to respond to real-time events. This delay in response can be critical in the case of rapidly evolving threats or attacks where an immediate response is required to avoid potential consequences. Limited energy consumption may impede the detection or reaction to such events, increasing the risk to the security of the system. These delays in detection or response can impact the overall reliability and security of the device in the face of persistent attacks or threats.
4. Impact on Encryption Algorithms. To ensure the security of IoT device data, encryption algorithms may be employed. However, in low energy consumption modes, their usage may be restricted, and less effective algorithms may be selected. This creates a risk of reducing the level of data protection, as the use of less reliable encryption methods can make data more vulnerable to attacks by malicious actors. Limited energy consumption can affect the efficiency of encryption in embedded devices. The compromise between energy savings and encryption efficiency can be a factor in increasing the vulnerability of devices to potential threats and cyber-attacks. In turn, the reduction in the level of data protection due to the use of less reliable encryption methods can complicate the recovery or protection of information in the event of attacks or unauthorized access to the device.
5. Low battery levels can significantly impact the effectiveness of cryptographic methods used to protect data. Cryptographic algorithms that demand substantial computational resources may operate unstably or lose efficiency due to limited energy supply. This can lead to a reduction in the speed or accuracy of applying cryptographic methods, diminishing the level of data protection. With low battery charges, a device may lack sufficient power to effectively implement complex encryption algorithms, resulting in increased data processing times or even a decrease in the level of protection. Such unstable operation of cryptographic methods can compromise the security of the device, making it more vulnerable to attacks.

6. Recovery after power loss. Restoring the operation of an embedded IoT device to its correct functional state can be challenging following a power loss. This is because, during sudden shutdowns or disconnections, the device may lose information about its previous state and current data. The difficulty or even impossibility of returning to the previous state directly affects its reliability and functionality.

Let's consider the effectiveness of protection against attacks when using an encryption algorithm, where efficiency is denoted as E, the battery level is B, and the type of cryptographic methodology is C. One of the possible models of efficiency has the form of a linear function:

$$E = m \cdot B + c \cdot C, \tag{3}$$

where m and c are parameters reflecting the influence of the battery level and the type of cryptographic methodology, respectively.

Let's assume the values of the coefficients are as follows: $m = 0.5$ and $c = 0.8$. The battery level (B) varies from 1 to 10, and the cryptographic methodology parameter (C) can take values of 1 or 2. The possible values of data protection efficiency (E), calculated using model (1) and these parameters, are presented in table 1.

Table 1
Evaluation of the energy-based attack protection efficiency model for an embedded device.

Charge level B	Protection efficiency E	
	C=1	C=2
1	0.5·1+0.8·1=1.3	0.5·1+0.8·2=2.1
2	0.5·2+0.8·1=1.8	0.5·2+0.8·2=2.6
3	0.5·3+0.8·1=2.3	0.5·3+0.8·2=3.1
4	0.5·4+0.8·1=2.8	0.5·4+0.8·2=3.6
5	0.5·5+0.8·1=3.3	0.5·5+0.8·2=4.1
6	0.5·6+0.8·1=3.8	0.5·6+0.8·2=4.6
7	0.5·7+0.8·1=4.3	0.5·7+0.8·2=5.1
8	0.5·8+0.8·1=4.8	0.5·8+0.8·2=5.6
9	0.5·9+0.8·1=5.3	0.5·9+0.8·2=6.1
10	0.5·10+0.8·1=5.8	0.5·10+0.8·2=6.6

The linear model (3) can be adapted to more complex dependencies, following the example of a quadratic model:

$$E = a \cdot B^2 + b \cdot C^2 + d \cdot B \cdot C + e, \tag{4}$$

where a, b, d, e are coefficients reflecting the interaction of the battery level and the type of cryptographic methodology on the effectiveness of data protection.

Models (3), (4) can be supported and refined through experiments, data analysis, and parameter tuning, taking into account the influence of various factors on the effectiveness of data protection at specific battery levels and specific types of cryptographic methodologies.

5.3. Risks due to limited computational power

Cryptographic protection algorithms, in general, are quite complex and resource-intensive in terms of the computational resources of their technical platform. Therefore, insufficient computational power of IoT devices has several consequences for their security:

1. Limited capacity for strong encryption application. The incompatibility of the computational resources of the embedded device with the requirements of strong, computationally complex encryption algorithms creates a risk of resorting to weaker encryption methods. This limitation may compel the device to choose less resource-intensive computational methods, which, in turn, may have lower resistance to cyberattacks.

2. Authentication failure due to resource constraints. Computational limitations can diminish the suitability of an embedded device for implementing robust identity verification methods, such as biometric data or complex encryption algorithms, thereby increasing vulnerability to attacks. Additionally, the limited memory of embedded devices can complicate the storage and management of authentication-related data, such as passwords, keys, or ciphers. This may lead to the use of less secure methods for storing identification information or a reduction in the number of available authentication methods. Therefore, the challenge of implementing proper authentication in embedded devices is associated with both the potential complexity of authentication algorithms and ensuring secure processes for storing and managing identity information. Moreover, the constraint on computational power may negatively impact the authentication process itself, resulting in the implementation of slower or less reliable authentication processes. The limited speed of the embedded device in processing authentication requests can make them less responsive to user requests in real-time or increase response times. Overall, the rejection of robust authentication methods decreases the device's level of protection.

6. Cryptographic models for risk analysis

In the context of security for embedded IoT devices, a key aspect is considering their resource constraints. These constraints directly impact the effectiveness of implementing security mechanisms and strategies. It is important to realize that each type of constraint – whether it's memory, battery charge, or energy consumption – poses unique challenges and requires specific solutions [26]. As the analysis shows, memory, battery charge, and energy consumption constraints significantly influence the cryptographic protection of information in IoT devices (table 2).

Table 2
Impact of embedded device resource constraints on information security.

Type of constraint	Impact on information security
Memory limitation	Complicates storage and management of encryption keys. Limits resources available for access control and authentication.
Battery charge constraint	Creates the risk of unpredictable interruptions in the device's operation. Reduces cryptography efficiency due to low battery charge.
Limited power consumption	Leads to a transition to low-power mode, restricting the use of powerful encryption algorithms. Affects response speed to threats due to standby mode for energy conservation.

Memory limitations often impact the device's ability to store encryption keys and other essential data, increasing the risk of unauthorized access and information leakage.

Meanwhile, battery charge limitations may lead to unforeseen disruptions in the device's operation, reducing its reliability and the effectiveness of protective mechanisms. Finally, limited energy consumption can restrict the application of resource-intensive protective algorithms, particularly in the field of cryptographic security.

Each of these aspects requires detailed consideration and analysis to ensure effective and adequate protection for embedded IoT devices.

6.1. Memory constraints

Memory constraints in IoT devices can pose a significant risk to data security. On one hand, limited memory can complicate the storage of large amounts of data or complex software algorithms necessary for effective cryptographic protection. On the other hand, insufficient memory can reduce the efficiency of key management, which is critically important for ensuring the security of communication processes. Memory limitations in IoT devices can lead to inadequate storage and management of encryption keys, increasing vulnerability to attacks.

The degree of impact of memory constraints on key storage, security management, and system vulnerabilities is illustrated in the diagram (figure 1). It is based on a conceptual analysis of the impact of memory constraints on these security aspects of IoT devices. The percentages indicated on the diagram reflect widely accepted expert estimates in the field of IoT cybersecurity, based on their experience and analysis of current trends in IoT technology development. These data do not represent specific quantitative research but rather provide a general understanding of trends in the field.

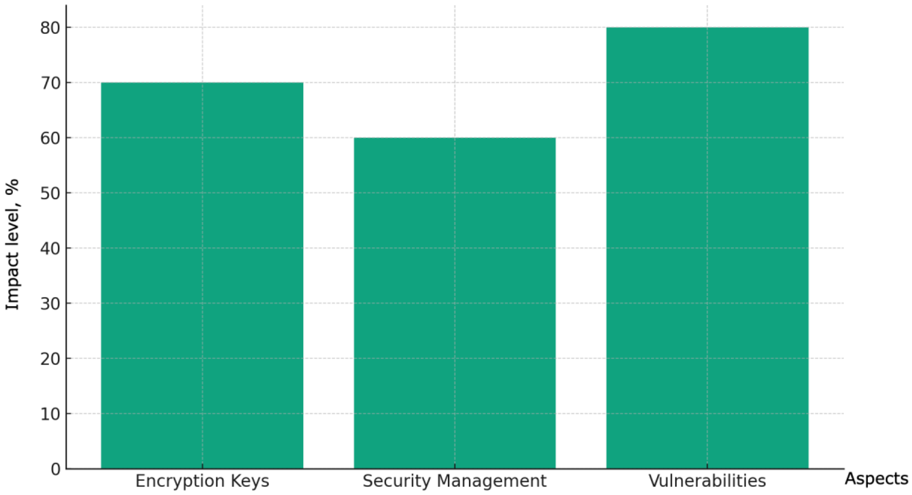


Figure 1: Impact of device memory constraints on its security.

6.2. Battery charge limitations

Battery charge limitations in IoT devices can cause disruptions in their operation, especially in critical situations. This may lead to a failure to perform essential security operations and unauthorized access to data. Additionally, a low battery charge can limit the effectiveness of encryption and other protective mechanisms. The limited battery life of IoT devices can result in unexpected shutdowns or reduced security functionality, increasing the risk of data leaks.

Let's define a function that relates the battery charge level to the runtime of security protocols. Let B be the initial battery charge level, and T be the duration of security algorithm operation in hours. Then:

$$T = a \cdot \ln ln(B) + b, \tag{5}$$

where a and b are constants based on the energy consumption characteristics of the device.

The diagram (figure 2) illustrates the impact of battery charge limitations on the activity of security protocols, the risk of data loss, and the constraints of protective mechanisms. This diagram is developed based on a qualitative analysis of the effects that battery charge limitations may have on the security aspects of IoT devices. The percentages on the diagram reflect estimated conclusions derived from theoretical considerations and expert opinions in this field, emphasizing the importance of considering energy aspects in the development of protective strategies for IoT. The diagram shows that battery charge limitations have the most significant impact on the risk of data loss during interruptions in operation. This underscores the importance of developing energy-efficient solutions to ensure the reliability and continuity of security functions.

6.3. Limitations on energy consumption

Limitations on energy consumption in IoT devices can be an obstacle to using resource-intensive security algorithms, especially in the field of cryptographic protection. This may lead to the selection of less powerful, and therefore less secure, encryption algorithms. Additionally, limited energy can

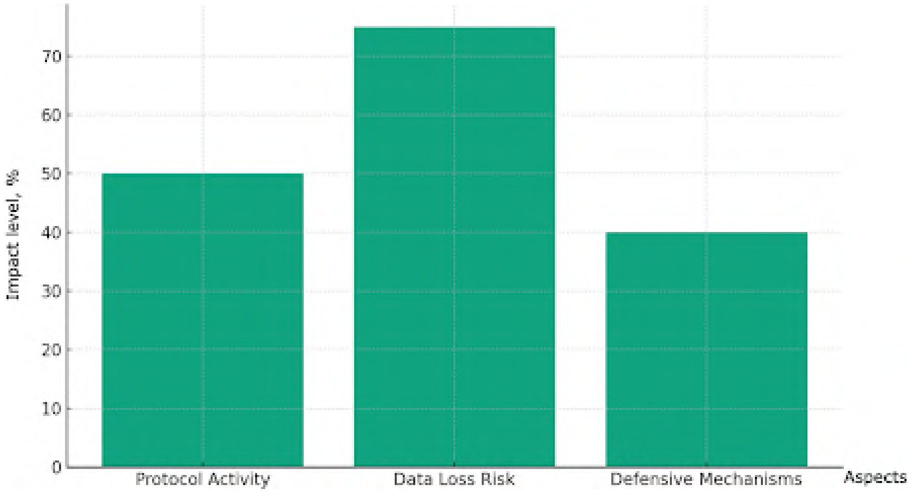


Figure 2: Impact of battery charge limitations on device security.

slow down the processes of detecting and responding to potential cyber threats. The difficulty of using complex cryptographic algorithms in IoT devices makes them vulnerable to advanced cyber-attacks.

Let's model the efficiency of cryptographic algorithms in relation to energy consumption. Let E represent the effectiveness of the applied algorithm's security properties, and P represent energy consumption. Then, the efficiency of cryptographic algorithms can be described by a polynomial function:

$$E = c_1 \cdot P^2 + c_2 \cdot P + c_3, \tag{6}$$

where c_1, c_2 and c_3 are coefficients determined based on the computational capabilities of the device.

The figure 3 depicts the diagram of the impact of energy consumption constraints on the security of IoT devices. The data for this diagram were formulated based on expert discussions and an assessment of potential consequences of limited energy consumption on the protective mechanisms of IoT devices. The percentage indicators reflect the generalized expert opinion on the importance of this aspect in the context of the development and application of cryptographic security systems. The diagram shows that limited energy consumption most significantly affects the selection and effectiveness of secure algorithms. This emphasizes the need for the development of energy-efficient cryptographic solutions that can provide an adequate level of security with constrained energy consumption.

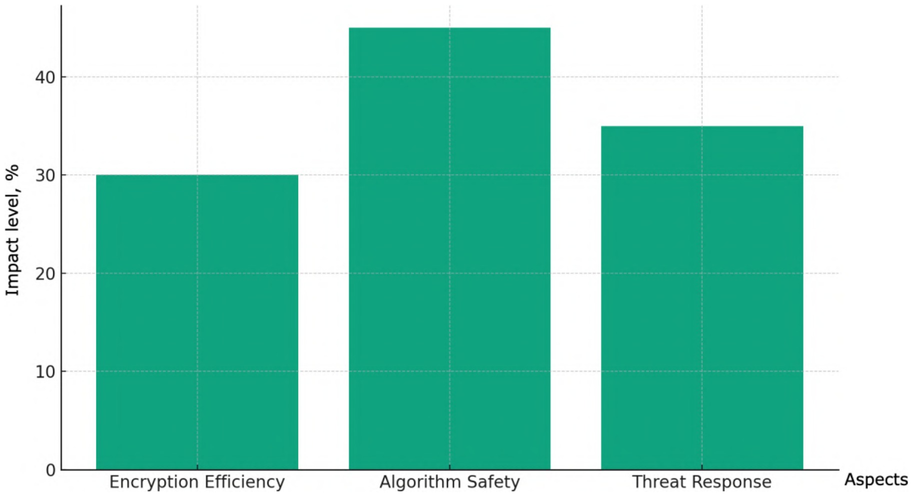


Figure 3: Impact of device energy consumption constraints on its security.

7. Strategies for optimizing security in IoT devices with limited resources

In the context of ensuring the security of IoT devices, optimizing their limited resources becomes crucial. This requires an innovative approach that takes into account both technical constraints and security needs. By focusing on key aspects of such limitations, such as memory, battery charge, and energy consumption, effective strategies can be developed to enhance the security level of IoT systems.

1. Lightweight Cryptographic Algorithms: Development and use of cryptographic algorithms that require minimal resources for execution but still provide reliable data protection.
2. Efficient Energy Management Algorithms: Implementation of algorithms that optimize energy consumption without compromising security can ensure longer device runtime and security system reliability.
3. Memory Usage Optimization: Development of methods for efficient utilization of limited memory space, including compact storage of encryption keys and using memory for security functions.
4. Adaptive Security Mechanisms: Creation of security systems capable of adapting to changing resource constraints to maintain an optimal level of protection in different operating conditions.
5. Improvement of Authentication Algorithms: Implementation of effective authentication algorithms that provide a high level of security with limited computational resources.
6. Secure Communication Protocols: Development of specialized communication protocols for IoT that are optimized for efficient resource utilization and ensure reliable data protection.

These strategies form the foundation for ensuring the security of IoT devices operating in resource-constrained environments. They enable a balance between security needs and constraints in memory, energy consumption, and operational resources, providing effective protection against potential threats.

8. Discussion

In light of the presented analysis, it is crucial to delve deeper into the discussion of the perspectives for further research in the field of IoT security with constrained resources. One of the key directions is the development and implementation of more efficient algorithms that consider the specificity of IoT devices. This includes not only technical aspects but also taking into account the diversity of applications of IoT devices in various industries.

The need for improvement in cryptographic protection methods is evident, especially in the context of limited memory and computational resources. The development of lightweight yet robust cryptographic algorithms can be key to enhancing overall security. Additionally, there is a necessity to develop flexible and adaptive security systems capable of effectively operating under resource constraints and quickly adapting to new threats and challenges.

Significant potential lies in research on energy-efficient technologies for IoT devices. Energy is a critical resource for many IoT systems, so developing methods for efficient energy management can significantly increase the autonomy and reliability of devices. It is also essential to consider the interaction between different components of IoT systems to optimize overall efficiency and security.

9. Conclusions

In the context of ensuring security for IoT devices with limited resources, it is important to recognize that effective security requires a multidimensional approach. This approach should involve the integration of technical innovations and strategic planning. Considering the constraints in memory, power consumption, and computational power, the development of lightweight cryptographic algorithms that utilize minimal resources becomes a priority to ensure reliable data protection.

Adapting security systems to the changing operational conditions of IoT devices is another crucial aspect. Security systems should be flexible, adaptive, and capable of maintaining a high level of security despite resource limitations. This includes not only technical aspects but also operational resource management, especially concerning energy and memory.

Innovations in authentication algorithms and energy-efficient technologies are essential for enhancing the autonomy and reliability of IoT devices. Further research in these areas should focus on developing solutions that can efficiently operate under resource constraints while providing reliable protection against current and future cyber threats.

Given the rapid advancement of technologies and the constant growth of cyber threats, continuous updating and adaptation of security mechanisms are integral parts of a security assurance strategy. Updating security solutions in response to new threats will help maintain a high level of protection while expanding the possibilities of applying IoT technologies in various domains.

10. Authors contribution

The authors confirm contribution to the paper as follows: study conception and design: I. Rozlomii, A. Yarmilko; data collection: I. Rozlomii; analysis and interpretation of results: I. Rozlomii, A. Yarmilko, S. Naumenko; draft manuscript preparation: I. Rozlomii, A. Yarmilko, S. Naumenko. All authors reviewed the results and approved the final version of the manuscript.

References

[1] S. Maitra, K. Yelamarthi, Rapidly Deployable IoT Architecture with Data Security: Implementation and Experimental Evaluation, Sensors 19 (2019) 2484. doi:10.3390/s19112484.

[2] Y. B. Shapovalov, Z. I. Bilyk, S. A. Usenko, V. B. Shapovalov, K. H. Postova, S. O. Zhadan, P. D. Antonenko, Harnessing personal smart tools for enhanced STEM education: exploring IoT integration, Educational Technology Quarterly 2023 (2023) 210–232. doi:10.55056/etq.604.

[3] O. V. Klochko, V. M. Fedorets, M. V. Mazur, Y. P. Liulko, An IoT system based on open APIs and geolocation for human health data analysis, CTE Workshop Proceedings 10 (2023) 399–413. doi:10.55056/cte.567.

[4] P. M. Chanal, M. S. Kakkasageri, Security and Privacy in IoT: A Survey, Wireless Personal Communications 115 (2020) 1667–1693. doi:10.1007/s11277-020-07649-9.

[5] N. Balyk, S. Leshchuk, D. Yatsenyak, Design and implementation of an IoT-based educational model for smart homes: a STEM approach, Journal of Edge Computing 2 (2023) 148–162. doi:10.55056/jec.632.

[6] S. Deep, X. Zheng, A. Jolfaei, D. Yu, P. Ostovari, A. K. Bashir, A survey of security and privacy issues in the Internet of Things from the layered context, Transactions on Emerging Telecommunications Technologies 33 (2022) e3935. doi:10.1002/ett.3935.

[7] N. M. Lobanchykova, I. A. Pilkevych, O. Korchenko, Analysis and protection of IoT systems: Edge computing and decentralized decision-making, Journal of Edge Computing 1 (2022) 55–67. doi:10.55056/jec.573.

[8] K. Yang, D. Blaauw, D. Sylvester, Hardware Designs for Security in Ultra-Low-Power IoT Systems: An Overview and Survey, IEEE Micro 37 (2017) 72–89. doi:10.1109/MM.2017.4241357.

[9] S. Shen, K. Zhang, Y. Zhou, S. Ci, Security in edge-assisted Internet of Things: challenges and solutions, Science China Information Sciences 63 (2020) 220302. doi:10.1007/s11432-019-2906-y.

[10] A. I. Jony, A. K. B. Arnob, A long short-term memory based approach for detecting cyber attacks in IoT using CIC-IoT2023 dataset, Journal of Edge Computing (2024). doi:10.55056/jec.648.

[11] F. Meneghello, M. Calore, D. Zucchetto, M. Polese, A. Zanella, IoT: Internet of threats? A survey of practical security vulnerabilities in real IoT devices, IEEE Internet of Things Journal 6 (2019) 8182–8201. doi:10.1109/JIOT.2019.2935189.

[12] M. Ammar, G. Russello, B. Crispo, Internet of Things: A survey on the security of IoT frameworks, Journal of Information Security and Applications 38 (2018) 8–27. doi:10.1016/j.jisa.2017.11.002.

[13] X. Jiang, M. Lora, S. Chattopadhyay, An experimental analysis of security vulnerabilities in industrial IoT devices, ACM Transactions on Internet Technology 20 (2020) 1–24. doi:10.1145/3379542.

[14] Rachit, S. Bhatt, P. R. Ragiri, Security trends in Internet of Things: A survey, SN Applied Sciences 3 (2021) 121. doi:10.1007/s42452-021-04156-9.

[15] M. Yu, J. Zhuge, M. Cao, Z. Shi, L. Jiang, A Survey of Security Vulnerability Analysis, Discovery, Detection, and Mitigation on IoT Devices, Future Internet 12 (2020) 27. doi:10.3390/fi12020027.

[16] B. D. Davis, J. C. Mason, M. Anwar, Vulnerability Studies and Security Postures of IoT Devices: A Smart Home Case Study, IEEE Internet of Things Journal 7 (2020) 10102–10110. doi:10.1109/JIOT.2020.2983983.

[17] R. T. Tiburski, C. R. Moratelli, S. F. Johann, M. V. Neves, E. de Matos, L. A. Amaral, F. Hessel, Lightweight Security Architecture Based on Embedded Virtualization and Trust Mechanisms for IoT Edge Devices, IEEE Communications Magazine 57 (2019) 67–73. doi:10.1109/MCOM.2018.1701047.

[18] S. Rajesh, V. Paul, V. G. Menon, M. R. Khosravi, A Secure and Efficient Lightweight Symmetric Encryption Scheme for Transfer of Text Files between Embedded IoT Devices, Symmetry 11 (2019) 293. doi:10.3390/sym11020293.

[19] B. C. Chifor, I. Bica, V. V. Patriciu, F. Pop, A security authorization scheme for smart home internet of things devices, Future Generation Computer Systems 86 (2018) 740–749. doi:10.1016/j.future.2017.05.048.

[20] M. Parmar, P. Shah, Internet of things-blockchain lightweight cryptography to data security and integrity for intelligent application, International Journal of Electrical and Computer Engineering (IJECE) 13 (2023) 4422–4431. doi:10.11591/ijece.v13i4.pp4422-4431.

[21] M. A. Jabraeil Jamali, B. Bahrami, A. Heidari, P. Allahverdizadeh, F. Norouzi, IoT Architecture, in: Towards the Internet of Things: Architectures, Security, and Applications, Springer International Publishing, Cham, 2020, pp. 9–31. doi:10.1007/978-3-030-18468-1_2.

[22] E. A. Shammar, A. T. Zahary, The Internet of Things (IoT): a survey of techniques, operating systems, and trends, Library Hi Tech 38 (2020) 5–66. doi:10.1108/LHT-12-2018-0200.

[23] S. I. Al-Sharekh, K. H. Al-Shqeerat, Security Challenges and Limitations in IoT Environments, IJCSNS International Journal of Computer Science and Network Security 19 (2019) 193–199. URL: http://paper.ijcsns.org/07_book/201902/20190224.pdf.

[24] O. L. Korenivska, V. B. Benedytskyi, O. V. Andreiev, M. G. Medvediev, A system for monitoring the microclimate parameters of premises based on the Internet of Things and edge devices, Journal of Edge Computing 2 (2023) 125–147. doi:10.55056/jec.614.

[25] T. M. Nikitchuk, T. A. Vakaliuk, O. A. Chernysh, O. L. Korenivska, L. A. Martseva, V. V. Osadchyi, Non-contact photoplethysmographic sensors for monitoring students' cardiovascular system functional state in an IoT system, Journal of Edge Computing 1 (2022) 17–28. doi:10.55056/jec.570.

[26] I. Rozlomii, A. Yarmilko, S. Naumenko, P. Mykhailovskyi, IoT Smart Implants: Information Security and the Implementation of Lightweight Cryptography, CEUR Workshop Proceedings 3609 (2023) 145–156. URL: https://ceur-ws.org/Vol-3609/paper12.pdf.

Automated Internet of Things system for monitoring indoor air quality

Nataliia A. Kulykovska[1], Artur V. Timenko[1], Svitlana S. Hrushko[1] and Vadym V. Shkarupylo[2,3]

[1]*National University “Zaporizhzhia Polytechnic”, 64 Zhukovsky Str., Zaporizhzhia, 69063, Ukraine*
[2]*National University of Life and Environmental Sciences of Ukraine, 15 Heroyiv Oborony Str., Kyiv, 03041, Ukraine*
[3]*G.E. Pukhov Institute for Modelling in Energy Engineering of the National Academy of Sciences of Ukraine, 15 General Naumov Str., Kyiv, 03164, Ukraine*

Abstract

This article explores the potential of Internet of Things technologies in creating a comprehensive air quality monitoring system with an emphasis on the indoor environment. The goal is to improve the quality of energy devices and protect the environment by ensuring optimal air conditions. This study highlights the role of embedded sensors in creating a universal Internet of Things based monitoring system that responds to various control parameters while excluding extraneous data. Key factors including temperature, humidity, dust control, air quality and energy efficiency are considered as critical aspects affecting the performance and lifetime of electrical systems and devices. This paper proposes the integration of sensors in wireless networks and the development of data processing and analysis algorithms to en-sure accurate and efficient determination of air quality. The system structure is proposed, which consists of three main modules: device modules, data processing modules, and application modules. The device module contains sensors to measure various parameters, while the data processing module processes the sensor data and the application module visualizes the data in real time. A management decision-making algorithm is proposed, which guides users based on air quality indicators. The paper defines air quality criteria, including temperature, humidity, carbon dioxide levels, and particulate matter concentration. Monitoring of these parameters allows early detection of air pollution and prompt corrective measures. The Internet of Things system was tested with a range of sensors, Arduino boards and the Blynk Internet of Things platform. Sensor data is displayed in real-time and alerts are sent when values exceed acceptable limits. The proposed system is an effective solution for maintaining indoor air quality. In conclusion, this study proposes a practical Internet of Things based air quality monitoring system suitable for indoor environments.

Keywords

Internet of things, monitoring system, air quality, temperature, Arduino, Blynk

1. Introduction

In the contemporary world, characterized by rapid technological advancements, the field of air quality monitoring has emerged as both relevant and crucial [1]. The advent of the Internet of Things (IoT) has ushered in a plethora of opportunities to develop efficient monitoring tools that facilitate frequent assessments and immediate troubleshooting [2].

One of the key aspects of this solution is the versatility of drone sensors. The presence of these sensors in the IoT interface potentially allows for the creation of a monitoring system capable of responding to a wide range of random control parameters, while excluding extraneous operational data, allowing informed decisions about device operation and ensuring safety [3].

Moreover, the ambient air conditions within an enclosed space play a pivotal role in determining the performance and longevity of electrical systems and devices. Temperature control, as a primary concern, cannot be overstated [4]. Extreme temperatures, whether excessively hot or cold, can exert adverse effects on electronic components. Elevated temperatures can lead to overheating, causing

doors-2024: 4th Edge Computing Workshop, April 5, 2024, Zhytomyr, Ukraine

natalya.gontar@gmail.com (N. A. Kulykovska); timenko.artur@gmail.com (A. V. Timenko); grushko_ss@i.ua (S. S. Hrushko); shkarupylo.vadym@nubip.edu.ua (V. V. Shkarupylo)
0000-0003-4691-5102 (N. A. Kulykovska); 0000-0002-7871-4543 (A. V. Timenko); 0000-0002-0064-408X (S. S. Hrushko); 0000-0002-0523-8910 (V. V. Shkarupylo)

components to degrade and, ultimately, resulting in system failures. Conversely, lower temperatures can create conditions conducive to condensation and moisture-related damage [5].

Another critical factor is humidity levels, which have a substantial impact on electrical systems. Excessive humidity can accelerate corrosion and lead to short circuits, compromising the integrity of devices. Conversely, low humidity levels can induce static electricity buildup, posing a risk to sensitive components. To mitigate these risks, air conditioning systems are often equipped with humidity control features to maintain optimal conditions.

Furthermore, air conditioning systems play a crucial role in managing dust and particulate matter within the environment. Air filters integrated into these systems effectively remove dust, allergens, and particulates from the air, ensuring that devices remain clean and unobstructed by debris [6]. Without such filtration, dust buildup can impede ventilation and exacerbate overheating issues.

Air quality stands as another crucial factor influencing device performance. Subpar air quality, characterized by high pollutant levels, can accelerate wear and tear on devices, necessitating more frequent maintenance and potentially reducing their operational lifespan [7].

Lastly, energy consumption is a significant consideration. Air conditioning systems consume electricity to function, and their efficiency directly impacts energy consumption, subsequently affecting operational costs and environmental sustainability. Properly maintained and optimized air conditioning systems can effectively manage energy usage, thereby reducing both financial expenditures and environmental footprints [8].

In conclusion, it is evident that the indoor air environment significantly influences the condition and performance of electrical systems and devices. Temperature regulation, humidity control, dust management, air quality, and energy efficiency are all interconnected aspects of this relationship. Moreover, the integration of IoT systems for air quality monitoring enhances our ability to create and maintain a conducive environment for devices, ensuring their reliability and longevity while optimizing energy usage for a more sustainable future.

In this context, this scientific work aims to explore and analyze the potential of utilizing IoT technologies to create air quality monitoring systems. It encompasses the integration of sensors into wireless sensor networks and the development of data processing and analysis algorithms to enable accurate and efficient determination of air quality. The outcomes of this research have the potential to make a significant contribution to improving the quality of life and environmental protection.

2. Literature review

In the field of the IoT, one of the most promising areas is the development of systems for monitoring indoor air quality. Scientific publications in recent years have emphasized the importance of integrating advanced sensors to truly and accurately monitor various air pollutants such as particulate matter, volatile organic compounds, and carbon dioxide [9]. These innovations in sensor technology include the development of miniaturized, cost-effective and energy-efficient devices, facilitating their widespread adoption in various indoor environments such as homes, offices and manufacturing plants.

In parallel with technological innovation, much attention is being paid to the development of machine learning algorithms and data analysis methods to interpret the vast amount of information coming from sensors. These methods can predict air quality trends, identify sources of pollution, and develop strategies to eliminate them. However, big data processing and analysis pose a significant challenge, highlighting the need to develop robust algorithms that can efficiently process and provide actionable insights from sensor data[10].

Integrating IoT systems with existing building management systems and heating, ventilation and air conditioning (HVAC) systems is a key area of research. This integration aims to optimize air quality control and energy efficiency. At the same time, the problem of interoperability between different IoT devices and platforms remains relevant, and the development of universal standards and protocols is critical for the smooth integration of various IoT components [11].

An important area of research is the development of user interfaces and ensuring the accessibility of air quality monitoring systems [12]. Simplifying the user interface for non-experts and providing easy access to air quality information is important for wider adoption. The development of mobile applications and cloud platforms for monitoring and managing indoor air quality is also being actively explored [13].

3. Methods

It is proposed to create an information system for monitoring atmospheric air pollution based on the results of the analysis and selection of optimal solutions of the complex technologies of the IoT in accordance with the proposed structure of the system shown in figure 1, using IoT technologies.

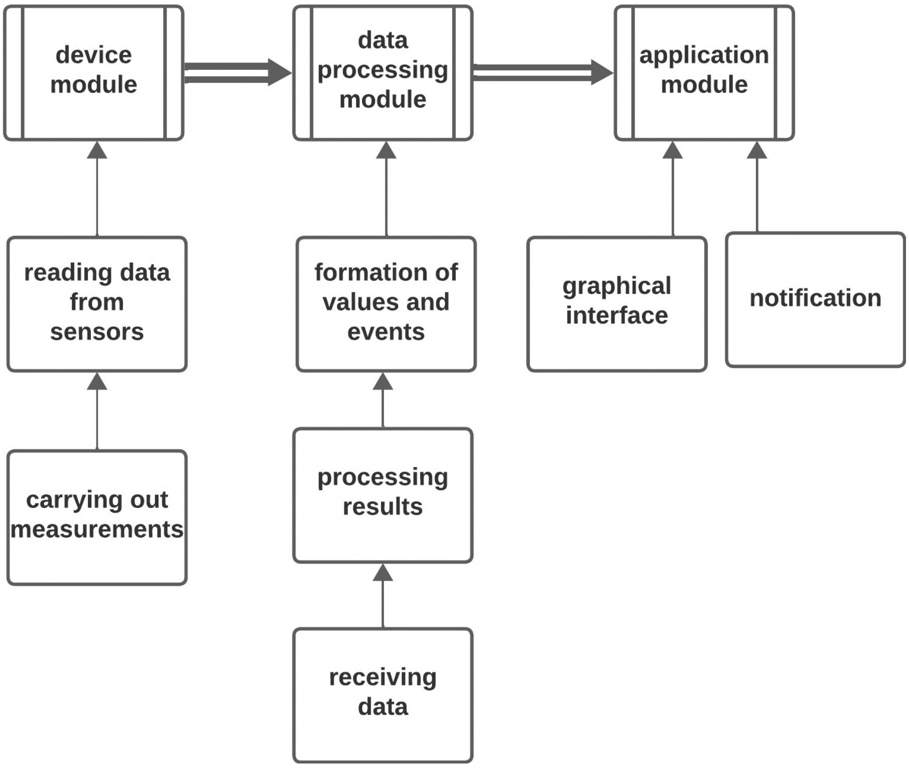


Figure 1: Structural diagram of the IoT system.

Based on this structure of the system, three main modules were selected, namely the device module, the data processing module and the application module.

- devices consist of a system control board, various sensors, auxiliary devices for emulating their operation, components for emulating the operation of a COM port for connecting external devices via the RS232 interface;
- processing modules consist of certain instructions in the code, which become valid sensor detection functions, read data from the sensors, process them according to this function and transfer them to the data visualization module for further display;
- application module displays the received data from the sensors, depending on their values will be circled in a certain color. After receiving the data, if the values of the sensors are above the limits of the accepted values, the system will notify the user about the actions that must be taken in order not to put yourself and your health at risk and to improve the air quality. This module uses the Blynk IoT software application and its built-in functionality to visualize the received data from the sensors in real time [14].

The proposed algorithm consists of six main steps (figure 2):

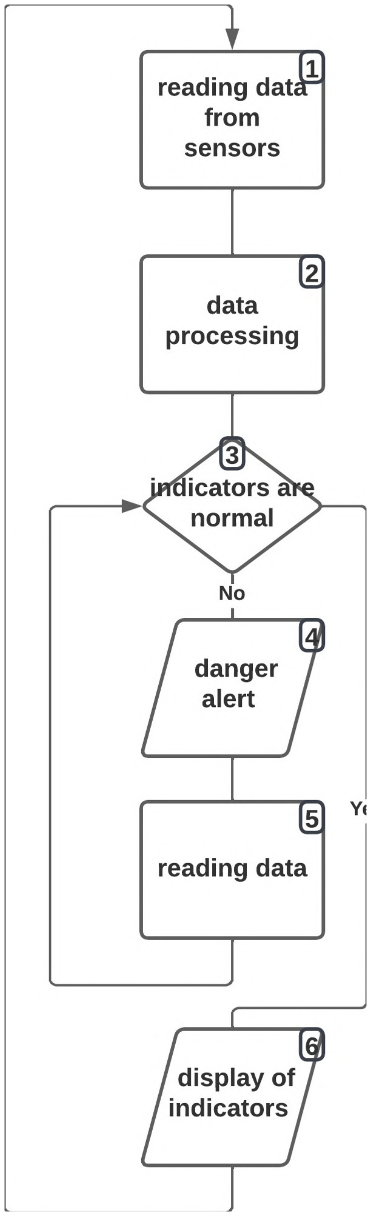


Figure 2: Work algorithm.

- item 1 – formation of a request;
- item 2 – subsequent processing of data in accordance with the secure section of the value;
- item 3 – if the indicators are normal, data on the consumer's sale in a convenient form, go to item 6;
- item 4 – if the data indicators do not correspond to the safe part, the user will be sent an agreement with recommendations, transition to item 5;
- item 5 – verification of changes in data indicators;
- item 6 – the user receives data from the sensors in a convenient form in the application.

The following indicators were selected for the air quality monitoring system:

- air temperature can affect people's comfort and quality of life. It can also affect substances dissolved in the wind and their mobility;
- air humidity can affect people's sense of comfort. Air with low levels of humidity can cause discomfort and negatively affect health and quality of life. High humidity can also promote the growth of fungi and mold;

- carbon dioxide (CO_2). Elevated levels of carbon dioxide in the air can be harmful to human health, causing shortness of breath and other problems. They can also serve as an indicator of the decrease in air quality over time;
- dust particles PM2.5 and PM10. These particles in the air can be very dangerous to health, then can penetrate deep into the respiratory tract and expand various lung and heart diseases. Monitoring the levels of these solution particles detects air pollution in time and takes measures to improve it.

Accurate measurement of local air quality limits is an aspect of ensuring a healthy and comfortable environment for living and working. This is especially relevant in megacities, where people spend most of their time in-doors. Measuring parameters such as CO_2 concentration, humidity level and air quality can identify your problems and help you take timely measures to solve them [15].

Table 1
The limit values of air quality criteria that can affect the human condition at all times.

Criteria	Indicators
Temperature	The range from 9.6 ℃ to 34.8 ℃ is chosen as a comfortable air temperature
Humidity	Humidity below 30% and above 60% is harmful, so 30% to 60% humidity was chosen as the safe range.
Carbon dioxide	At a concentration of carbon dioxide above 0.14% (1400 ppm), air quality is classified as low, i.e. values below 1400 ppm are considered normals
Particulate matter PM2.5	Safe range: less than 12 μg/m³ (micrograms per cubic meter).
Dust particles PM10	Safe range: less than 50 μg/m³ per day.

4. Testing of IoT system

After analyzing the necessary components for the operation of the air monitoring system, it was decided to add the following components to the device module:

- Arduino UNO board;
- the MQ135 sensor is a gas sensor that measures the concentration of various harmful gases in the air, in particular, ammonia, hydrogen sulfide, benzene and other gases;
- potentiometer (POT-HG) – an element used as a slider to change the values of the MQ135 sensor;
- the DHT11 sensor is a sensor that measures air temperature and humidity.
- another DHT11 sensor – emulation of PM2.5 and PM10 sensors;
- compel is a component that emulates a virtual COM port for connecting external devices via the RS232 interface.

The Blynk IoT system was chosen for dis-play and monitoring capabilities. It allows you to receive, store, and display data from sensors in real time. A Template was created, ac-cording to which a Device will be created, which will receive, store and display information from sensors.

To understand the ranges of acceptable values, a certain color was set, for example, if at the moment the sensor value is in the nor-mal range, the scale next to it will be green, if it is in the range of values that are above or below the norm, it will be red. The temperature from 0 ℃ to 9.6 ℃ and 34.8 ℃ to 60 ℃ will be considered harmful, so the scale around it will be red, if the temperature value is from 9.6 ℃ to 34.8 ℃ – green. Similar actions were taken for the data of other sensors [14].

The trigger frequency for events was set to 1 minute. This means that as long as the sensor value is in the range of bad values, an event will be triggered every minute and sent to the message block

(Timeline) of the device. The free version of Blynk stores these messages for 1 week, but the paid version can store them longer.

As a result, the system reads data from sensors in real time and displays them on the panel. When the values are above or below the norm, notifications are sent to the message bar about what should be done to change the air quality.

As a result of the analysis, it was decided to use a mobile application for system control (figure 3).

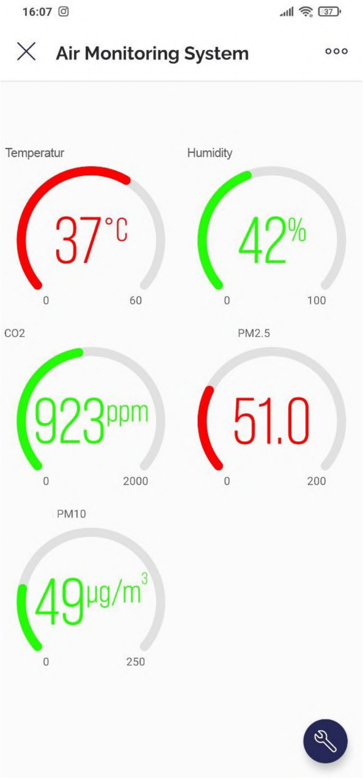


Figure 3: Mobile application for system control.

In figure 4 shows graphs of changes in parameters: humidity , air temperature, carbon dioxide content, as well as the concentration of PM2.5 and PM10 particles during of this hour period (as an illustration). The graphs were constructed using the software described in this work and worked in real time.

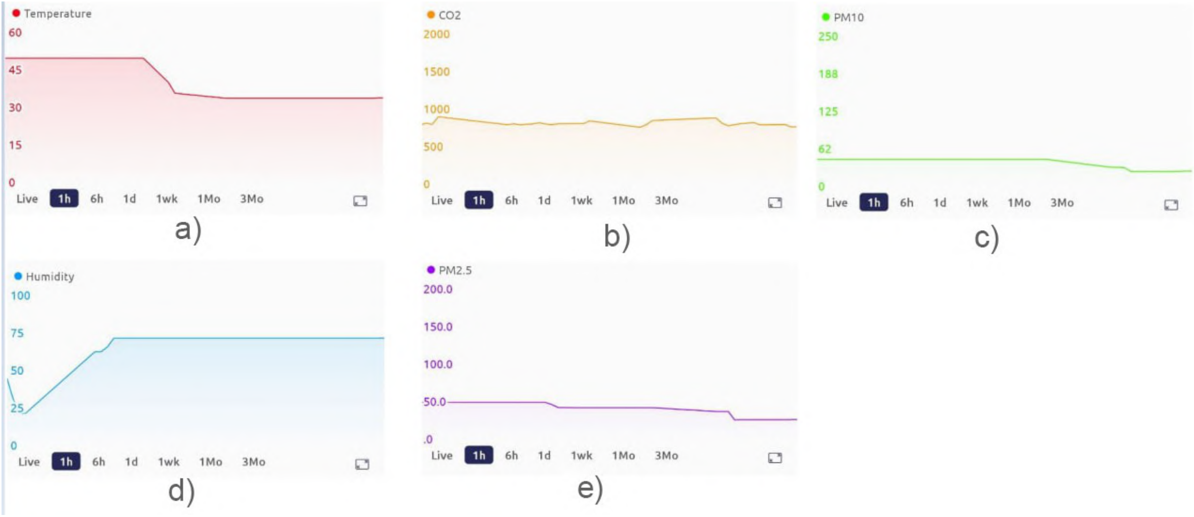


Figure 4: Graph of changes in air parameters.

5. Conclusion

The problem of air quality monitoring based on the automatic interaction of various devices that transmit data using the IoT technology is considered. In the course of the work, the structure of the system and the method of data analysis and visualization, processing results using IoT technologies such as Proteus and Blynk, as well as other tools, were developed.

Accurate measurement of the limits of indoor air quality criteria is an important aspect to ensure a healthy and comfortable environment for the operation of devices and work. Measurement of parameters such as CO_2 concentration, humidity level and air quality can identify potential problems and facilitate the adoption of timely measures to solve them.

A system analysis and justification of the choice of software and technical solutions, which are necessary for the implementation of this system and all its stages, were carried out. The use of technologies such as Arduino, Proteus and Blynk IoT made it possible to develop a real-time air quality monitoring system within the modern IoT concept. This developed system can be used as a prototype for organizing monitoring in changing environments and responding to various critical situations.

6. Author contributions

N. Kulykovska and A. Timenko conceived the idea and designed the system architecture; S. Hrushko analyzed the data processing methods; N. Kulykovska and V. Shkarupylo performed the simulations and analyzed the results. All authors discussed the results, contributed to the final manuscript, and approved the submitted version.

References

[1] IoT Editorial Office, Acknowledgment to the Reviewers of IoT in 2022, IoT 4 (2023) 56–56. doi:10.3390/iot4010003.

[2] J. K. Verma, D. K. Saxena, V. G.-P. Díaz, V. Shendryk, Cloud IoT: Concepts, Paradigms, and Applications, Chapman and Hall/CRC, 2022. doi:10.1201/9781003155577.

[3] T. M. Nikitchuk, T. A. Vakaliuk, O. A. Chernysh, O. L. Korenivska, L. A. Martseva, V. V. Osadchyi, Non-contact photoplethysmographic sensors for monitoring students' cardiovascular system functional state in an IoT system, Journal of Edge Computing 1 (2022) 17–28. doi:10.55056/jec.570.

[4] N. Balyk, S. Leshchuk, D. Yatsenyak, Design and implementation of an IoT-based educational model for smart homes: a STEM approach, Journal of Edge Computing 2 (2023) 148–162. doi:10.55056/jec.632.

[5] I. Klymenko, A. Haidai, C. Nikolskyi, V. Tkachenko, Architectural concept of the monitoring system based on the iot neural module of data analytics, Adaptive automatic control systems 2 (2022) 111–123. doi:10.20535/1560-8956.41.2022.271355.

[6] I. A. Pilkevych, D. L. Fedorchuk, M. P. Romanchuk, O. M. Naumchak, Approach to the fake news detection using the graph neural networks, Journal of Edge Computing 2 (2023) 24–36. doi:10.55056/jec.592.

[7] J. D. Hagar, IoT Test Design: Frameworks, Techniques, Attacks, Patterns, and Tours, in: IoT System Testing: An IoT Journey from Devices to Analytics and the Edge, Apress, Berkeley, CA, 2022, pp. 153–164. doi:10.1007/978-1-4842-8276-2_10.

[8] R. Herrero, Analytical model of IoT CoAP traffic, Digital Communications and Networks 2 (2019) 63–68. doi:10.1016/j.dcan.2018.07.001.

[9] V. Upadrista, The IoT Standards Reference Model, in: IoT Standards with Blockchain: Enterprise Methodology for Internet of Things, Apress, Berkeley, CA, 2021, pp. 61–86. doi:10.1007/978-1-4842-7271-8_4.

[10] M. Chen, Y. Miao, I. Humar, Large Sensor Network OPNET Model Debugging, in: OPNET IoT Simulation, Springer Singapore, Singapore, 2019, pp. 249–294. doi:10.1007/978-981-32-9170-6_4.

[11] K. K. Bhardwaj, A. Khanna, D. K. Sharma, A. Chhabra, Designing Energy-Efficient IoT-Based Intelligent Transport System: Need, Architecture, Characteristics, Challenges, and Applications, in: M. Mittal, S. Tanwar, B. Agarwal, L. M. Goyal (Eds.), Energy Conservation for IoT Devices : Concepts, Paradigms and Solutions, Springer Singapore, Singapore, 2019, pp. 209–233. doi:10.1007/978-981-13-7399-2_9.

[12] M. Albano, A. Skou, L. L. Ferreira, T. Le Guilly, P. D. Pedersen, T. B. Pedersen, P. Olsen, L. Šikšnys, R. Smid, P. Stluka, C. Le Pape, C. Desdouits, R. Castiñeira, R. Socorro, I. Isasa, J. Jokinen, L. Manero, A. Milo, J. Monge, A. Zabasta, K. Kondratjevs, N. Kunicina, Application system design - energy optimisation, in: J. Delsing (Ed.), IoT Automation: Arrowhead Framework, CRC Press, Boca Raton, 2017, pp. 211–246. doi:10.1201/9781315367897-8.

[13] N. Singh, S. Kumar, B. K. Kanaujia, A New Trend to Power Up Next-Generation Internet of Things (IoT) Devices: 'Rectenna', in: M. Mittal, S. Tanwar, B. Agarwal, L. M. Goyal (Eds.), Energy Conservation for IoT Devices : Concepts, Paradigms and Solutions, Springer, Singapore, 2019, pp. 331–356. doi:10.1007/978-981-13-7399-2_14.

[14] N. Kulykovska, A. Timenko, S. Hrushko, M. Ilyashenko, A Semantic Chatbot for Internet of Things Management, in: 2022 IEEE 9th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T), 2022, pp. 246–250. doi:10.1109/PICST57299.2022.10238683.

[15] M. Mittal, S. C. Pandey, The Rudiments of Energy Conservation and IoT, in: M. Mittal, S. Tanwar, B. Agarwal, L. M. Goyal (Eds.), Energy Conservation for IoT Devices : Concepts, Paradigms and Solutions, Springer, Singapore, 2019, pp. 1–17. doi:10.1007/978-981-13-7399-2_1.