УДК 004
DOI https://doi.org/10.32689/maup.it.2025.1.8

Денис ДУБОВИК
кандидат технічних наук, IT expert KIVI MEDIA, denic3d@gmail.com
ORCID: *0000-0002-8983-0646*

Тетяна ДУБОВИК
старший викладач кафедри спеціалізованих комп'ютерних систем,
Навчально-наукового інституту «Український державний хіміко-технологічний університет», Український державний університет науки і технологій,
tetyana_dubovik@udhtu.edu.ua
ORCID: *0000-0002-2359-2569*

Вадим МАТУС
магістр кафедри Спеціалізованих комп'ютерних систем
Навчально-наукового інституту «Український державний хіміко-технологічний університет», Український державний університет науки і технологій,
udhtu@udhtu.edu.ua
ORCID: *0009-0007-8473-2741*

РОЗРОБКА ПРОГРАМНОЇ ПЛАТФОРМИ ДЛЯ КЕРУВАННЯ МАНІПУЛЯТОРОМ З ПІДТРИМКОЮ WIFI ЗА ДОПОМОГОЮ КАМЕРИ

Анотація. Метою даної роботи *є створення зручного та ефективного програмного інтерфейсу для управління маніпулятором, встановлення зв'язку з камерою для забезпечення візуального зворотного зв'язку, а також розробка модульної системи, здатної вміщати в себе різні плагіни.*

Методологія. *У статті розглядаються ключові особливості програмної платформи та її архітектури, акцентуючи увагу на взаємодії між маніпулятором, камерою та різними плагінами. Маніпулятор, у своїй основній функції як виконавчий механізм, оснащений затисками для роботи та камерою над ними (використана кінематична схема, задача зворотної кінематики). Смартфон виступає сервером (підключають по Wi-Fi), відправляє команди маніпулятору. Програма на сервері має модульну структуру: в будь-який момент часу вона використовує лише один плагін, який через інтерфейс отримує дані від маніпулятора, аналізує їх та, залежно від функціональності плагіну, надає відповідні команди маніпулятору. Плата управління маніпулятором має протокол взаємодії з сервером, що дозволяє писати контролер на різних мовах програмування на боці сервера. Графічний інтерфейс програми розроблений на мові програмування Kotlin з використанням графічного фреймворку Jetpack Compose. Це дозволяє імпортувати програму на різні платформи, такі як Linux, Mac OS та Windows. Наразі програма працює лише на платформі Android.*

Наукова новизна. *Важливість використання вхідних даних з відеокамери в управлінні роботом, що дозволяє виконувати такі завдання, як розпізнавання об'єктів, відстеження та покращення прийняття рішень, технічні аспекти програми, включаючи протоколи обміну даними для маніпулятора та камери, архітектуру плагінів та інтерфейс користувача. Особливості модульної програмної платформи для управління маніпулятором з відеопотоком. Модульність, гнучкість і відеовхід в режимі реального часу сприяють його практичності і актуальності в сфері робототехніки.*

Практичне значення. *Модульна програмна платформа для управління маніпулятором з реалізацією відеопотоку на Android призначена для спрощення керування роботизованими маніпуляторами за допомогою єдиного програмного інтерфейсу та інтеграції відео в режимі реального часу для покращення контролю та прийняття рішень цієї роботи полягає в її потенційному застосуванні в різних галузях, таких як виробництво, охорона здоров'я та автоматизація. Модульний характер платформи спрощує інтеграцію конкретних функцій, роблячи її придатною для широкого спектру завдань у сфері робототехніці.*

Актуальність *платформи полягає в зростаючому попиті на гнучкі роботизовані системи, які можна адаптувати під різні сфери застосування. Забезпечуючи модульність і розширюваність, платформа дозволяє розробникам створювати та інтегрувати конкретні плагіни для різних завдань без необхідності перепрограмування всієї системи управління.*

Висновки. *Основні результати досліджень свідчать, що модульна програмна платформа успішно забезпечує єдиний інтерфейс для управління маніпулятором та інтеграції відеопотоку в режимі реального часу. Робот протестовано. Гнучкість платформи дозволяє користувачам адаптувати систему до різних завдань, що робить її цінним інструментом для дослідників і розробників у галузі робототехніки.*

Ключові слова: *роботизований маніпулятор, програмна платформа, відеопотік, модульність, взаємодія, автоматизація.*

Denys DUBOVYK, Tetiana DUBOVYK, Vadym MATUS. DEVELOPMENT OF A SOFTWARE PLATFORM FOR CONTROLLING A WIFI-ENABLED MANIPULATOR USING A CAMERA

Abstract. The purpose of this work *is to create a convenient and effective program interface for controlling the manipulator, establishing communication with the camera to provide visual feedback, as well as developing a modular system that can accommodate various plugins.*

Methodology. *The article discusses the key features of the software platform and its architecture, focusing on the interaction between the manipulator, camera, and various plugins. The manipulator, in its main function as an actuator, is equipped with clamps for work and a chamber above them (a kinematic scheme is used, a problem of reverse kinematics). The smartphone acts as a server (connected via Wi-Fi), sends commands to the manipulator. The program on the server has a modular structure: at any given time it uses only one plugin, which receives data from the manipulator through the interface, analyzes it and, depending on the functionality of the plugin, provides the appropriate commands to the manipulator. The program is developed in the Kotlin programming language using the Jetpack Compose graphics framework. It allows you to import the program to various platforms such as Linux, Mac OS, and Windows. Currently, the app only works on the Android platform.*

Scientific novelty. *The importance of using input from a video camera in robot control, allowing tasks such as object recognition, tracking and improving decision-making, technical aspects of the application, including data exchange protocols for the manipulator and camera, plugin architecture, and user interface. Features of the modular software platform for controlling a manipulator with a video stream. Modularity, flexibility and real-time video input contribute to its practicality and relevance in the field of robotics.*

The practical significance *of the Modular Manipulator Control Software Platform with Video Stream Implementation on Android is designed to simplify the control of robotic manipulators through a single software interface and real-time video integration to improve control and decision-making This work lies in its potential application in various industries such as manufacturing, healthcare, and automation. The modular nature of the platform simplifies integration of specific functions, making it suitable for a wide range of tasks in the field of robotics.*

The relevance of *the platform lies in the growing demand for flexible robotic systems that can be adapted to different applications. By providing modularity and extensibility, the platform allows developers to create and integrate specific plugins for different tasks without having to reprogram the entire control system.*

Conclusions. *The main results of the research indicate that the modular software platform successfully provides a single interface for manipulator control and real-time video stream integration. The platform's flexibility allows users to adapt the system to different tasks, making it a valuable tool for researchers and developers in the field of robotics.*

Key words: *Robotic manipulator, software platform, video stream, modularity, interaction, automation.*

Аналіз досліджень та публікацій. В роботі [1] авторами виконано комплексний аналіз прямих та обернених кінематичних розрахунків для просторового маніпулятора з новою архітектурою на шість ступенів вільності.

В роботі [4] автори зосередились на проблемі оберненої кінематики для гібридного (паралельно-серійного) маніпулятора з п'ятьма ступенями вільності (5-DOF). Проблема оберненої кінематики полягає у визначенні керуючих впливів на приводи, необхідних для виконання певної траєкторії руху в системі керування. У статті спершу надається короткий опис структури маніпулятора, яка включає в себе 3 ступені вільності для паралельної частини та 2 ступені вільності для серійної частини, і потім пояснюється алгоритм розв'язання оберненої кінематики. Запропонований алгоритм використовує формулу добутку експонент (PoE) для застосування до еквівалентного маніпулятора зі структурою серійного типу. Отриманий алгоритм надає рішення в закритій формі, не роблячи припущень щодо геометрії маніпулятора. Він був підтверджений результатами кейс-дослідження. Метод розв'язання проблеми оберненої кінематики може бути адаптований для інших гібридних маніпуляторів.

Автор [9] розглянув важливу проблему в галузі робототехніки: вирішення оберненої кінематичної задачі. Це завдання необхідне для керування робототехнічними системами і передбачає визначення керуючих дій для виконання заданої траєкторії руху. Дослідження фокусується на гібридному (паралельно-послідовному) маніпуляторі з п'ятьма ступенями вільності. Основним досягненням дослідження є розробка алгоритму, який надає аналітичне рішення без будь-яких припущень щодо геометрії маніпулятора.

Виклад основного матеріалу. Розроблений маніпулятор, у своїй основній функції як виконавчий механізм, оснащений робочим органом з затисками для роботи та камерою над ними. Смартфон виступає сервером (підключають по Wi-Fi), відправляє команди маніпулятору. Останній може лише отримувати параметри серводвигунів, встановлювати кути обертання, швидкість та інші параметри. На боці сервера працює програма, яка обмінюється даними з маніпулятором по Wi-Fi і надає програмний інтерфейс для взаємодії з ним.

Програма на сервері має модульну структуру: в будь-який момент часу вона використовує лише один плагін, який через інтерфейс отримує дані від маніпулятора, аналізує їх та, залежно від функціональності плагіну, надає відповідні команди маніпулятору.

Ця модульність дозволяє виконувати різноманітні завдання за допомогою маніпулятора, не потребуючи постійного прошивання контролера плати управління маніпулятором.

Плата управління маніпулятором має протокол взаємодії з сервером, що дозволяє писати контролер на різних мовах програмування на боці сервера [9].

Графічний інтерфейс програми розроблений на мові програмування Kotlin з використанням графічного фреймворку Jetpack Compose. Це дозволяє імпортувати програму на різні платформи, такі як Linux, Mac OS та Windows. Наразі програма працює лише на платформі Android.

Програмний інтерфейс для створення плагінів.

Для забезпечення контролю передачі команд управління та даних між плагінами та маніпулятором, було створено програмний інтерфейс (API). Цей інтерфейс регулює параметри команд, які передаються від плагінів до маніпулятора, щоб уникнути випадкових значень обертів серводвигунів та інших параметрів.

Крім того, API включає у себе корисні функції, які спрощують розробку плагінів для програми, забезпечуючи можливість не переписувати різні функції, такі як пряма та зворотна кінематика. Ці функції були реалізовані один раз у межах інтерфейсу та спрощують написання плагінів.

Розробники плагінів можуть (за бажанням) використовувати виключно високорівневі функції для управління маніпулятором, такі як зсув робочого органу вліво чи вправо та інші. Крім того, API дозволяє контролювати та реєструвати різні дії плагінів, що надає більше можливостей для контролю плагінів як розробникам програм, так і розробникам плагінів. Цей інтерфейс також має методи для отримання відеопотоку з маніпулятора для подальшого аналізу та визначення поведінки маніпулятора.

Для створення власного плагіну необхідно створити папку з файлом «plugin.kt» всередині неї. У цьому файлі потрібно створити клас, який реалізує інтерфейс «Plugin». Після цього цю папку потрібно розмістити в папці «plugin» основної програми. Після цього система автоматично розпізнає цей плагін, і його можна буде використовувати.

Інтерфейс камери містить в собі протокол обміну даними між камерою та комп'ютером, та всі необхідні методи для зручного отримання зображення с камери. Схема інтерфейсу камери зображена на рисунку 1.

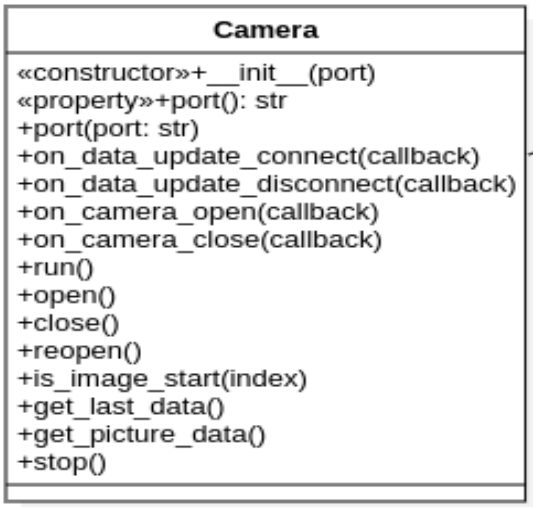

Рис. 1. Інтерфейс камери

Камера функціонує через окремий потік для уникнення блокування основного процесу при отриманні даних з послідовного порту. Крім того, використовується паттерн Singleton для обмеження створення лише одного екземпляру класу, що оптимізує отримання даних з послідовного порту.

Для отримання зображення з камери програміст повинен створити екземпляр класу та встановити порт, до якого підключена камера. Після цього можна отримати зображення за допомогою методу «get_picture_data», який поверне масив значень пікселів чорно-білого зображення. Альтернативно, можна автоматично отримувати нові зображення після їх отримання, налаштувавши метод зворотного виклику, який буде викликатися інтерфейсом камери та передавати дані нового зображення.

Створення об'єкту та налаштування правильного порту автоматизовано програмою. Відповідно, користувач може призначити потрібний порт через вікно налаштувань. Створені об'єкти інтерфейсу камери та інтерфейсу маніпулятора (про нього далі) стають доступні для плагінів через інтерфейс класу «Plugin», який має успадкувати кожен плагін.

Для сприяння зручності розробки графічної частини плагінів, також розроблено віджет камери «CameraWidget». Структура класу зображена на діаграмі, наведеній на рисунку 2

Інтерфейс маніпулятора. Як і задача інтерфейсу камери так і інтерфейсу маніпулятора, в інкапсуляції роботи протоколу обміну даними між маніпулятором та комп'ютером, та дати розробнику плагінів всі необхідні методи для управління маніпулятором.

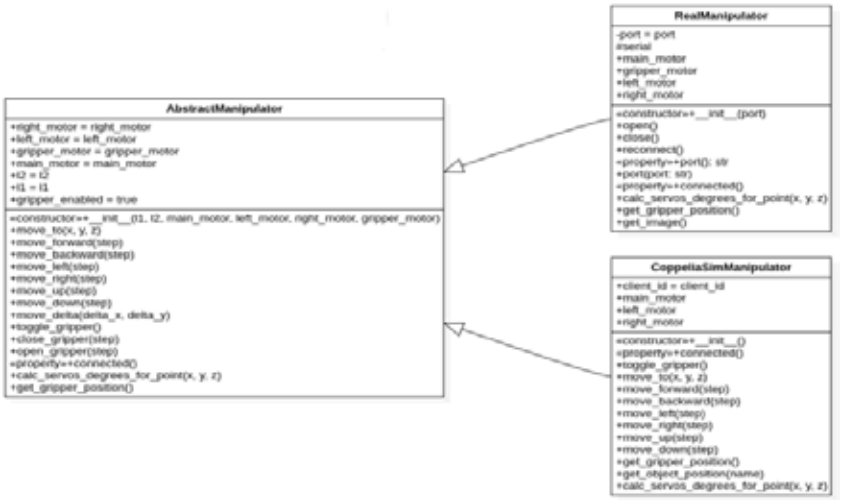

Рис. 2. Інтерфейс маніпулятора

Інтерфейс фізичного маніпулятору реалізує «RealManipulator», для швидкої розробки також була використана віртуальне середовище розробки, та для управління віртуальним маніпулятором був створений інтерфейс «CoppeliaSimManipulator» [2, 10].

Інтерфейс дає розробнику високо рівневі методи для управління маніпулятором, такі як, пересування захвату маніпулятора по всім осям з заданим кроком, або переміщення захвату в задану позицію на координатній площині відносно маніпулятора. Як і з інтерфейсом камери, створенням об'єкту та встановленням вірного порту займається сама система, до яких отримати доступ можливо через інтерфейс «Plugin».

Ручний контроль

Розроблені два плагіни для прикладу роботи модульної системи. Ручний контроль передбачає функціонал для ручного керування маніпулятором, такі функції як:

1) Поворот вліво
2) Поворот вправо
3) Переміщення захвату вперед
4) Переміщення захвату назад
5) Переміщення захвату вгору
6) Переміщення захвату вниз
7) Замикання захвату
8) Розмикання захвату
9) Покрокове замикання захвату
10) Покрокове розмикання захвату

Робоча область плагіну зображена на рисунку 3.

Рис. 3. Повзунки відповідають за дії маніпулятора, які на них написані

Розпізнавання обличчя. Плагін який реалізує розпізнавання обличчя з відео-потоку маніпулятора, виконує розпізнавання та автоматичне переміщення маніпулятора в залежності від напрямку обличчя (див. на рисунку 4), обличчя яке в данний момент знаходиться під трекінгом – буде завжди в центрі кадру, так як мапулятор буде переміщюватись разом з особою.

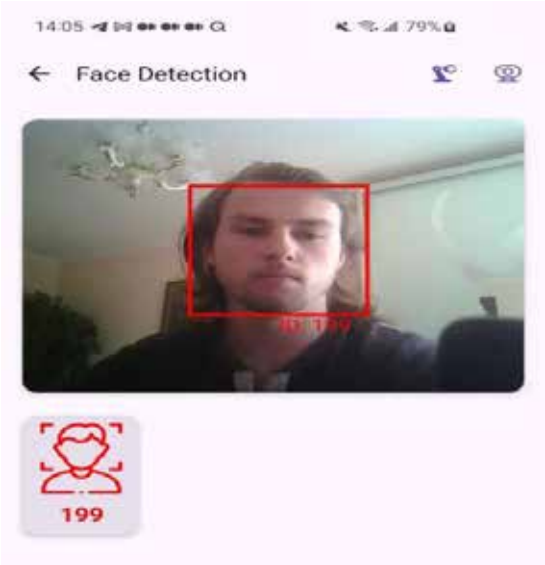

Рис. 4. Розпізнавання обличчя

Активне обличчя трекінгу підсвічується червоною рамкою, також знизу список всіх розпізнаних обличь, картики клікабельні та мають ID, після натискання на картку вона стає активною для відслідковування, якщо в кадрі буде декілька обличь, то маніпулятор буде слідкувати тільки за вибраним. Приклад з декількома обличчями зображений на рисунку 5.

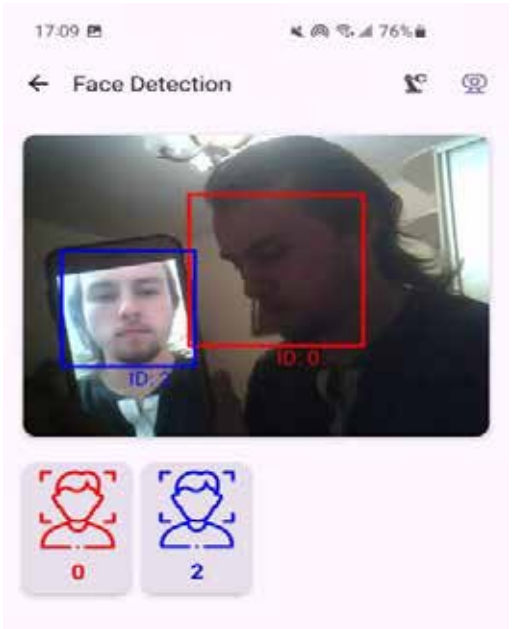

Рис. 5. Розпізнавання декількох обличь

Схеми підключення сервоприводів до шилду плати управління зображена на рисунку 6 та підключення ESP32-CAM до Arduino Uno [6,8] на рисунку 7.

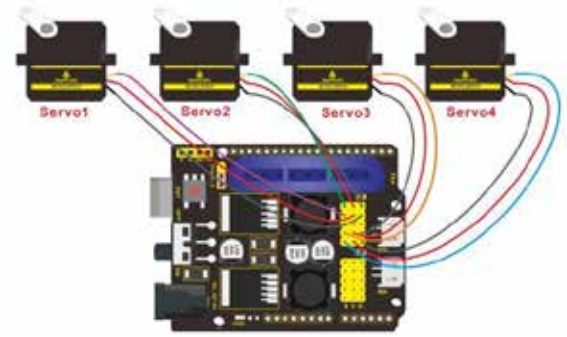

Рис. 6. Схема підключення сервоприводів до шилду плати управління

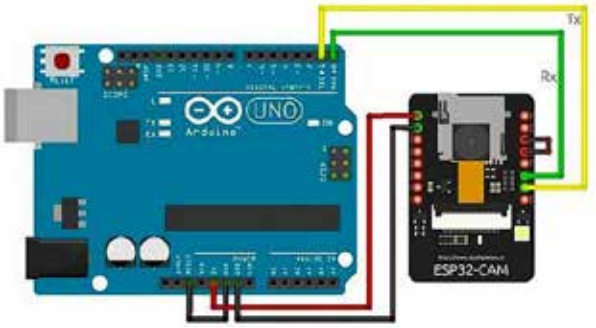

Рис. 7. Схема підключення ESP32-CAM до Arduino Uno

Підключення до маніпулятора відбувається через WiFi до ESP32-CAM, яка в свою чергу передає команди по Serial порту Arduino UNO, яка керує сервоприводами [10].

Фрагмент коду програми:

Лістинг А.3 – Текст файлу «ControlViewModel.kt»

```
class ControlViewModel(
    private val manipulator: WsManipulator,
    private val camera: Camera,
) : ViewModel() {
    val manipulatorState = manipulator.connectionState
        .onEach {
            if (it is ManipulatorState.Disconnected) disconnectCamera()
        }
        .stateIn(
            scope = viewModelScope,
            started = SharingStarted.WhileSubscribed(5_000),
            initialValue = ManipulatorState.Disconnected("")
        )
    val cameraState = camera.connectionState
        .onEach {
            if (it is CameraState.Disconnected) disconnectManipulator()
        }
```

Результати натурних іспитів

Плагін, що реалізує розпізнавання обличчя з відео-потоку маніпулятора, виконує розпізнавання та автоматичне переміщення маніпулятора в залежності від напрямку обличчя, обличчя яке в данний момент знаходиться під трекінгом – буде завжди в центрі кадру, так як маніпулятор буде переміщюватись разом з особою. Активне обличчя трекінгу підсвічується червоною рамкою, після натискання на картку вона стає активною для відслідковування, якщо в кадрі буде декілька обличь, то маніпулятор буде

слідкувати тільки за вибраним. Тестування системи проводилось в різних умовах, якість розпізнавання обличчя зведено в табл. 1.

Таблиця 1

Якість розпізнавання обличчя

Освітлення обличчя			**Швидкість руху маніпулятора з камерою**			**Розташування обличчя відносно камери**	
Денне світло	*Тінь на обличчі*	*Темно*	*Середня*	*Повільно*	*Висока*	*Ліворуч/праворуч або вгору/донизу*	*Напроти камери*
100%	70%-80%	15%	60%	100 %	25%	Не розпізнано	100%

Висновки. Для розробки було розраховано пряме та зворотне завдання кінематики, та динаміки робота-маніпулятора, сформована його математична модель, внаслідок чого з'явилась можливість провести дослідження математичної моделі робота-маніпулятора. В ході розробки та дослідження Android додатку для управляння WiFi-маніпулятором з камерою було розроблено:

1) прошивку для маніпулятора з протоколом для зв'язку з ним;

2) програмний інтерфейс для зв'язку з камерою та маніпулятором;

3) комплексні функції в складі інтерфейсу – які спрощують розробку програм управління маніпулятором;

5) крос-платформний та адаптивний графічний інтерфейс користувача для управління плагінами;

6) систему плагінів для розширення функціоналу програми;

7) графічний інтерфейс для управління маніпулятором та плагінами, інтерфейс адаптивний та крос-платформний, має всі необхідні функції для управління маніпулятором;

8) функціонуючу систему, яка може бути використана для різних галузях для різних цілей.

Систему протестовано при різних впливах на об'єкт (світло, нахил, швидкість, тощо) та визначено: правильність взаємодії робота з його підсистемами;

продуктивність робота в конкретних умовах; захищеність робота від зовнішніх впливів та його стійкість до помилок та збоїв; точність і швидкість розпізнавання об'єктів за допомогою камер; оцінка здатності робота визначати своє місцезнаходження та слідувати заданому маршруту.

В даний час розроблена система актуальна, та має перспективу на модернізацію та удосконалення. Може бути використана в різних сферах, де затребувана роботизація виробництва.

Список використаних джерел:

1. Golla P., Ramesh S., Bandyopadhyay S. (2023). URL: https://www.scopus.com/record/display.uri?eid=2-s2.0-85162178085&origin=resultslist

2. Katupitiya, Jayantha, Bentley, Kim DrivingMotors DC &Stepper. In: Interfacingwith

3. Lu Y., Hu B., Sun T. (2009). URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-67651153037&doi=10.1017%2fS0263574708004918&partnerID=40&md5=b6b76bdc269b31c9e03b94690f538e40

4. Nachtwey P. Build Better Machines Through Optimized Motion Control. SAE International, 2002. https://doi.org/10.4271/2002-01-1344

5. Quinn A.R.J., Saxby D.J., Yang F., de Sousa A.C.C., Pizzolato C. (2023). URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85151450848&doi=10.1016%2fj.jbiomech.2023.111557&partnerID=40&md5=97ebaabaec0f7bffbfb4f7f0422910c2

6. Sergeyeva O. V., Robuck E. A., Pavlenko N. Y., Dubovik T. N. Applications of computer systems based on the Arduino microprocessor system in chemical laboratory. The Eurasian Union of Scientists (ESU), 2019. 12 (69 (5)), 34–38.

7. Sergeyeva O., Lisenko V., Dubovik T., Patalakha M. "Development of a Wi-Ficontrolled mobile video device on the Arduino NANO basis",Eastern-European Journal of Enterprise Technologies, 2020. Vol. 3/9 (105), 55–60. DOI: https://doi.org/10.15587/1729-4061.2020.206558

8. Timmis H. "Mature Arduino Engineering.Making an Alarm System Using the Arduino", Practical Arduino Engineering. Apress. 2011. 197, doi: https://doi.org/10.1007/978-1-4302-3886-7_8

9. Tze Fun Chan., Keli Shi. Applied Intelligent Control of Induction Motor Drives. John Wiley & Sons (Asia) Pte Ltd., 2011. https://doi.org/10.1002/9780470825587

10. Wen K., Harton D., Laliberte T., Gosselin C. 2019. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85071484243&doi=10.1109%2fICRA.2019.8793772&partnerID=40&md5=d36896e17c6e27ad8710120530434881