



NURE

Kharkiv National University
of Radioelectronics

**Невлюдов І. Ш., Андрусевич А. О., Євсєєв В.В.,
Новоселов С.П., Демська Н.П.**



ПРОЄКТУВАННЯ МОБІЛЬНИХ МАНІПУЛЯЦІЙНИХ РОБОТІВ

Рекомендовано Науково-технічною радою Харківського національного університету радіоелектроніки (протокол № 4 від 27 травня 2022 року)

Рецензенти:

Замірець М. В., доктор технічних наук, професор, Заслужений діяч науки і техніки України, Лауреат Державної премії України в галузі науки і техніки, Директор ДП «Науково-дослідний технологічний інститут приладобудування»

Косенко В. В., доктор технічних наук, професор, помічник директора по науковій роботі ДП «Південний державний проектно-конструкторський та науково-дослідний інститут авіаційної промисловості»

Невлюдов І. Ш., Андрусевич А. О., Євсєєв В. В., Новоселов С. П., Демська Н. П. **Проектування мобільних маніпуляційних роботів**: Монографія. – Х. :, 2022. – 427 с.

У монографії проводиться аналіз сучасних мобільних маніпуляційних роботів, кінематика бортового маніпулятора на мобільному роботі, розглядається приклад силового розрахунку мобільного маніпулятора. Розглянуті способи розв'язання задачі врівноваження мобільного маніпулятора, залежно від маси корисного вантажу для переміщення. Проведено детальне дослідження сучасних одноплатних комп'ютерів, а саме конструкцію та архітектуру мікрокомп'ютера Raspberry Pi, як бази для розроблення системи управління, особливості роботи та налаштування шини GPIO. Показано принципи реалізації віддаленого доступу до управління роботом за допомогою IoT. Особлива увага приділена установці та налаштуванню бібліотеки OpenCV, розглянута послідовність виконання скриптів. Розглянуто шляхи реалізації методу сенсibilізації маніпулятора, за рахунок підключення різноманітних датчиків. Автори навели приклад не тільки програмної реалізації, але й показали схематичні рішення підключення апаратних засобів. Розроблена системи управління переміщенням мобільного робота, розроблено алгоритми управління переміщенням, на базі якого розроблено програмний скрипт управління, а також показано метод підключення та реалізації потокового відео з камери, яка дозволяє транслявати відео в режимі реального часу для візуалізації динамічного навколишнього середовища робота. На основі аналізу основних завдання реалізації системи розпізнавання та ідентифікації об'єктів була розроблена система ідентифікації та розпізнавання об'єктів у робочій зоні робота, з використанням бібліотеки Yolo та мови Python. Проведені дослідження методів керування групою мобільних роботів, для вирішення поставлених завдань в умовах техногенних катастроф. Основна увага приділена задачі пошуку оптимальної траєкторії і розглядають методи планування траєкторій на базі CNN. Розроблена програмно-апаратна архітектура системи планування для групи мобільних роботів. У результаті розроблена загальна системи управління мобільними роботами та проведенню результати математичного моделювання та експериментів. Наведено метод визначення траєкторії руху в невизначеному просторі, який реалізований у вигляді програмного модуля та проведено ряд експериментів, що показали правильність прийнятих рішень.

© Невлюдов І. Ш.
© Андрусевич А. О.
© Євсєєв В. В.
© Новоселов С. П.
© Демська Н. П.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....	7
ВСТУП.....	11
1 АНАЛІЗ СУЧАСНИХ МОБІЛЬНИХ МАНІПУЛЯЦІЙНИХ РОБОТІВ І ПРИНЦИПІВ ЇХ ФУНКЦІОНУВАННЯ	15
1.1 Сучасні мобільні маніпуляційні роботи	15
1.2 Механіка маніпуляторів мобільних роботів	23
1.2.1 Кінематичний аналіз механізму мобільного маніпулятора	24
1.2.2 Силловий розрахунок мобільного маніпулятора	32
1.2.3 Врівноважування мобільних маніпуляторів.....	34
1.3 Геометричний синтез маніпулятора для мобільного інформаційного робота на гусеничному шасі	36
1.3.1 Кінематика маніпулятора-трипода	45
1.3.2 Силловий аналіз маніпулятора.....	54
2 СИСТЕМИ КЕРУВАННЯ МОБІЛЬНИХ РОБОТІВ НА БАЗІ ОДНОПЛАТНИХ КОМП'ЮТЕРІВ ТА МІКРОПРОЦЕСОРІВ	61
2.1 Характеристики одноплатних комп'ютерів та мікропроцесорів	61
2.2 Asus Tinker Board	72
2.3 Raspberry Pi Foundation.....	75
2.4 Orange Pi.....	82
2.5 LattePanda	87
2.6 Odroid	88
2.7 PINE64	93
2.8. Модуль ESP32-CAM.....	96
2.9 Плати розширень, модулі	106
2.10 Датчики	125
3 АРХІТЕКТУРА ОДНОПЛАТНОГО КОМП'ЮТЕРА RASPBERRY PI 3 MODEL B PLUS	154

3.1 Основні елементи Raspberry Pi 3 Model B +.....	154
3.2 Електричні характеристики Raspberry Pi 3 Model B +.....	157
3.3 General Purpose Input Output (GPIO)	161
3.4 Послідовний інтерфейс I ² C	166
3.5 Послідовний інтерфейс SPI.....	170
3.6 Послідовний порт UART.....	173
4 ОПЕРАЦІЙНА СИСТЕМА RASPBIAN ДЛЯ РОБОТИ З OPENCV МОВОЮ PYTHON	176
4.1 OS Raspbian за допомогою NOOBS.....	176
4.2 Базові команди Linux для Raspberry Pi.....	180
4.3 OS Raspbian	186
4.4 Налаштування віддаленого доступу	188
4.4.1 Налаштування віддаленого доступу через VNC	188
4.4.2 Віддалений робочий стіл через Internet	195
4.5 Налаштування Samba на OS Raspbian	196
4.6 Створення резервної копії OS Raspbian.....	201
4.7 Установка бібліотеки OpenCV	204
4.8 Установка бібліотек для роботи з GPIO	212
5 ПІДКЛЮЧЕННЯ ТА ПРОГРАМУВАННЯ ДАТЧИКІВ МОВОЮ PYTHON.....	217
5.1 Підключення ультразвукового датчика HC-SR04	217
5.2 Підключення та програмування датчика температури і вологості DHT22	220
5.3 Підключення та програмування управління сервоприводами для системи комп'ютерного зору.....	227
6 РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ МОБІЛЬНОЮ ПЛАТФОРМОЮ НА БАЗІ МОВИ PYTHON	231
6.1 Скрипт управління DC двигунами мобільної платформи	231
6.2 Підключення і настройка відео-стріму з камери Raspbian Pi.....	233
6.3 Розробка алгоритмів керування мобільного робота	240

6.4 Розробка програми віддаленого керування мобільним роботом	250
6.5 Розробка клієнтської програми віддаленого керування мобільним роботом	252
6.6 Встановлення необхідних пакетів та налаштувань.....	255
7 РЕАЛІЗАЦІЯ СИСТЕМИ ІДЕНТИФІКАЦІЇ ОБ'ЄКТІВ МЕТОДОМ YOLO МОВОЮ PYTHON	260
7.1 Класифікація задач розпізнавання образів	260
7.2 Основні поняття побудови алгоритмів на базі CNN.....	263
7.3 Програмна реалізація систем ідентифікації об'єктів методом YOLO	271
7.4 Проведення експерименту ідентифікації об'єктів методом YOLO	277
8 ГРУПОВЕ УПРАВЛІННЯ МОБІЛЬНИМИ РОБОТАМИ	287
8.1 Проблематика групового управління у мобільній робототехніці.....	287
8.2 Завдання пошуку оптимальної траєкторії	289
8.3 Методика планування траєкторії за допомогою нейронної карти.....	302
8.3.1 Поняття нейронної картки.....	302
8.3.2 Архітектура системи планування траєкторії на базі нейронної мережі Хопфілда.....	303
8.3.3 Системні характеристики	304
8.4 Математична модель нейронної мережі Хопфілда	305
8.5 Розробка програмно-апаратної архітектури системи планування для групи мобільних роботів	309
9 РОЗРОБКА ЕКСПЕРЕМЕНТАЛЬНОГО МАКЕТУ МОБІЛЬНОГО МАНІПУЛЯЦІЙНОГО РОБОТА	316
9.1 Розробка системи керування мобільним маніпуляційним роботом	316
9.2 Розробка структури програмного забезпечення системи керування мобільним маніпуляційним роботом	332
9.3 Метод визначення траєкторії руху у невизначеному просторі	342
9.4 Розробка програмного модуля для моделювання переміщення мобільного робота	345

9.5 Розробка програмного модуля для керування маніпуляційним маніпулятором мобільного робота	347
9.6 Моделювання та експериментальні дослідження	354
ПЕРЕЛІК ПОСИЛАНЬ	356
ДОДАТОК А Розташування виводів Asus Tinker Board 2Гб на GPIO	405
ДОДАТОК Б Розташування виводів Raspberry Pi 2 &3 на GPIO	406
ДОДАТОК В Порівняльні характеристики основних моделей Raspberry	407
ДОДАТОК Г Розташування виводів SB LattePanda 2Гб.....	409
ДОДАТОК Д Порівняльна таблиця ODROID-C2 з аналогічними міні комп'ютерами.....	410
ДОДАТОК Е Програма сервісної частини дистанційного керування мобільним роботом.....	411
ДОДАТОК Ж Програмний засіб серверної частини системи дистанційного керування мобільного робота	413
ДОДАТОК И Скрипт Python для ультразвукового далекоміра.....	419
ДОДАТОК К Скрипт Python для одночасного управління двома сервомоторами.....	420

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

SBC	- (single-board computer) Одноплатні комп'ютери в ультра-маленькому форм-факторі;
PoE	- (Power over Ethernet) технологія, що дозволяє передавати віддаленого пристрою електричну енергію разом з даними через стандартну виту пару в мережі Ethernet;
USB	- (Universal Serial Bus) послідовний інтерфейс для підключення периферійних пристроїв до обчислювальної техніки;
GPIO	- (general-purpose input/output) інтерфейс для зв'язку між компонентами комп'ютерної системи, наприклад мікропроцесором і різними периферійними пристроями. Контакти GPIO можуть виступати як в ролі входу, так і в ролі виходу - це, як правило, конфігурується. GPIO контакти часто групуються в порти;
MicroUSB	- мініатюрні роз'єми для застосування USB в переносних і мобільних пристроях;
HDMI	- (High Definition Multimedia Interface) інтерфейс для мультимедіа високої чіткості, дозволяє передавати цифрові відеодані високої роздільної здатності та багатоканальні цифрові аудіосигнали із захистом від копіювання;
CSI	- інтерфейс для CSI камери;
DSI	- конектор для LCD дисплею;
PXE	- (Preboot eXecution Environment) завантаження ОС через мережевий інтерфейс використання локальних носіїв даних;
ZigBee	- специфікація мережевих протоколів регламентованих стандартом IEEE 802.15.4. ZigBee і IEEE 802.15.4 описують бездротові персональні обчислювальні мережі (WPAN) ;

- GPS - (Global Positioning System) супутникова система навігації, яка забезпечує вимірювання відстані, часу та визначає місце розташування у всесвітній системі координат WGS 84;
- GSM - (Global System for Mobile Communications) глобальний стандарт цифрового мобільного стільникового зв'язку з розділенням каналів за часом (TDMA) і частоті (FDMA) ;
- ШИМ (PWM) - (pulse-width modulation) процес управління потужністю методом пульсуючого включення і виключення приладу. Розрізняють аналогову ШИМ і цифрову ШИМ, двійкову (дворівневу) ШИМ і трійкову (трирівневу) ШИМ;
- SPI - (Serial Peripheral Interface) послідовний синхронний стандарт передачі даних в режимі повного дуплексу, призначений для забезпечення простого і недорогого високошвидкісного сполучення мікроконтролерів і периферії;
- SD - (Secure Digital Memory Card) формат карт пам'яті (флеш-пам'ять), розроблений SD Association (SDA) для використання в портативних пристроях;
- TFT - (thin-film transisto) рідкокристалічний дисплей з активною матрицею;
- I²C - (Inter-Integrated Circuit) послідовна асиметрична шина для зв'язку між інтегральними схемами всередині електронних приладів;
- SSH - (Secure Shell) мережевий протокол прикладного рівня, що дозволяє виробляти віддалене управління операційною системою і тунелювання TCP-з'єднань;
- SDRAM - (Synchronous Dynamic Random Access Memory) тип пристрою, що запам'ятовує, та використовується в комп'ютері і інших цифрових пристроях як ОЗУ;

- DDR3 - (double-data-rate three synchronous dynamic random access memory) це тип оперативної пам'яті, що використовується в обчислювальній техніці в якості оперативної і відеопам'яті;
- IoT - (internet of things) концепція обчислювальної мережі фізичних предметів, оснащених вбудованими технологіями для взаємодії один з одним або з зовнішнім середовищем, яка розглядає організацію таких мереж як явище, здатне перебудувати економічні та суспільні процеси, що виключає з частини дій і операцій необхідність участі людини;
- LPDDR3 - стандарт JESD209-3 «Low Power Memory Device Standard» пропонується більш висока швидкість обміну даними, збільшена енергоефективність і велика щільність пам'яті;
- CPU - (central processing unit) інтегральна схема (мікропроцесор), виконуюча машинні інструкції (код програм), головна частина апаратного забезпечення комп'ютера або програмованого логічного контролера;
- GPU - (graphics processing unit) інтегральна схема (мікропроцесор), для обробки і вирішення графічних завдань;
- FPS - (Frames Per Second) кількість кадрів в секунду на екрані монітора;
- eMMC - (embedded Multimedia Memory Card) це одночіповий накопичувач, найчастіше вживаний в компактних пристроях;
- DSI - (Display Interface Specifications) Специфікація визначає послідовну шину і протокол зв'язку між хостом (джерело зображення) і пристроєм (одержувачем зображення) ;
- DHCP - (Dynamic Host Configuration Protocol) протокол динамічної настройки вузла, що використовується для автоматичного динамічного присвоєння IP-адрес пристроїв мережі;
- КЗ - комп'ютерний зір;

SMB/CIFS - (Server Message Block) мережевий протокол прикладного рівня для віддаленого доступу до файлів, принтерів та інші ресурси мережі, а також для взаємодії між процесами;

RPi - Raspberry Pi ;

ПО - програмне забезпечення.

ВСТУП

Сучасний розвиток науки і техніки, призвів до створення нових технологій, які знайшли застосування в концепції Industry 4.0. Industry 4.0- це новий виток промислової революції, заснований на впровадженні передових досягнень у сферах штучного інтелекту, прийняття рішень, хмарних обчислень та хмарних даних, адитивних технологій, мікросистемної техніки тощо. Все це в сукупності дозволяє реалізовувати рішення для складних багаторівневих завдань у системі реального часу з урахуванням вимог, які висуваються до мобільних маніпуляційних роботів.

У першому розділі монографії проводиться аналіз сучасних мобільних маніпуляційних роботів, їх характеристик та конструктивних особливостей. Особлива увага приділяється кінематиці та механіці бортового маніпулятора на мобільному роботі, розглядається приклад силового розрахунку мобільного маніпулятора. Наведено геометричний синтез для розроблюваного мобільного маніпуляційного робота, з розширеними функціоналом, розглянуті способи вирішення задачі врівноваження мобільного маніпулятора в залежності від маси корисного вантажу для переміщення.

Другий розділ спрямований на розробку систем управління та освітлення мобільного маніпуляційного робота з розширеним функціоналом. Проведено детальне дослідження сучасних одноплатних комп'ютерів, спроектованих на базі мікроконтролерних та мікропроцесорних систем. Розглянуто особливості їхньої архітектури, характеристик та їх застосування з точки зору основи для розробки системи управління та прийняття рішень мобільного маніпуляційного робота з розширеними функціоналом. Також приділено увагу існуючим платам розширення, які підтримують шину GPIO, а також розглянуті датчики, які необхідні для надання відчуттів мобільного робота і захватного пристрою.

Третій розділ присвячений особливостям конструкцій та архітектурі одноплатного мікрокомп'ютера Raspberry Pi 3 model B plus. На базі нього буде

розроблено систему управління, проведено дослідження електричних характеристик, особливості роботи та налаштування шини GPIO. Проаналізовано особливості підключення та роботи з послідовними інтерфейсами: I2C, SPI, UART за допомогою яких буде здійснено підключення датчиків та систем надання відчуттів мобільного маніпуляційного робота з розширеними функціоналом.

Четвертий розділ присвячений дослідженню питань тонкого налаштування та встановлення операційної системи Raspbian за допомогою Noobs. Реалізація віддаленого доступу до робочого столу OS Raspbian виконана за допомогою технології IoT. У даному розділі автори приділяють особливу увагу установці та налаштуванню бібліотеки OpenCV, послідовності виконання скриптів та налаштувань шини GPIO. Це в сукупності дозволяє приступити до розробки алгоритмів та програмного забезпечення системи управління та прийняття рішень на мові Python.

У п'ятому розділі приділено увагу реалізації методу надання відчуттів мобільного маніпуляційного робота з розширеним функціоналом за рахунок підключення ультразвукових датчиків, датчиків вологості та температури. Всі перераховані датчики дозволяють вирішувати завдання контролю динамічного навколишнього середовища мобільного робота. Автори наводять приклад не тільки програмної реалізації, а й показують схематичні рішення підключення. Це дозволяє уникнути виходу з ладу датчика або Raspberry Pi 3 model B plus. Так само для збільшення кута огляду мобільного маніпуляційного робота з розширеним функціоналом в даному розділі наведено приклад підключення серводвигуна з встановленою на ньому камерою та фрагмент програмного коду для керування ним. Все це в сумі дозволяє збільшити кут огляду як по горизонталі так і по вертикалі.

В шостому розділі монографії увагу приділено розробці системи керування переміщенням мобільного маніпуляційного робота з розширеним функціоналом, розроблено алгоритми керування переміщенням, на базі яких розроблено програмний скрипт керування DC двигунами. Розроблено систему

віддаленого управління мобільним маніпуляційним роботом з розширеним функціоналом на базі «клієнт-серверної» архітектури, а також показано метод підключення та демонстрації потокового відео з камери, яка дозволяє транслювати відео в режимі реального часу для візуалізацій динамічного навколишнього середовища робота.

Сьомий розділ розглядає основні завдання реалізації системи розпізнавання та ідентифікації об'єктів на базі CNN. Автори в рамках даних досліджень пропонують свою класифікацію завдань розпізнавання, проводять дослідження особливостей побудови алгоритмів на базі CNN. Використовуючи бібліотеку Yolo та мову Python, була розроблена система ідентифікації та розпізнавання об'єктів у робочій зоні мобільного маніпуляційного робота з розширеним функціоналом, проведено низку експериментів щодо точності ідентифікації та розпізнавання об'єктів у залежності від кута огляду.

Восьмий розділ присвячено дослідженням у сфері групового управління мобільними маніпуляційними роботами з розширеним функціоналом, на вирішення поставлених завдань за умов техногенних катастроф. Розглянуто проблематику групового управління, виділено основні вузькі місця. Автори приділяють увагу задачі пошуку оптимальної траєкторії та розглядають методи планування траєкторії на базі CNN. Ґрунтуючись на математичній моделі HP Хопфілда, розроблена програмно-апаратна архітектура системи планування для групи мобільних маніпуляційних роботів з розширеним функціоналом.

Дев'ятий розділ присвячений розробці загальної системи управління мобільним маніпуляційним роботом з розширеними функціональними можливостями та проведенню математичного моделювання та експериментів. Запропоновано метод визначення траєкторії руху в недетермінованому просторі, який реалізований у вигляді програмного модуля. Проведено ряд експериментів, який показав правильність прийнятих рішень.

У монографії в ході проведення досліджень автори проводили аналіз сучасної наукової, науково-технічної літератури з міжнародних наукометричних баз даних, офіційної технічної документацій. Це дозволило не

тільки провести аналіз, а й застосувати на практиці деякі результати досліджень в рамках даної роботи.

Монографія буде корисна для вчених та фахівців, розробників мобільних маніпуляційних роботів, систем управління та систем прийняття рішень, а також магістрів у наступних галузях знань: 15 – Автоматизація та приладобудування за спеціальністю 151 – Автоматизація та комп'ютерно-інтегровані технології.

Монографія виконана в рамках наукових досліджень відповідно до державної теми №337 “ Інтелектуальна багатоцільова мобільна робототехнічна платформа з удосконаленими маніпуляційними можливостями ”

1 АНАЛІЗ СУЧАСНИХ МОБІЛЬНИХ МАНІПУЛЯЦІЙНИХ РОБОТІВ І ПРИНЦИПІВ ЇХ ФУНКЦІОНУВАННЯ

1.1 Сучасні мобільні маніпуляційні роботи

Існують різні погляди на те, як класифікувати роботів [1–3]. Розділяти їх на групи, в частині механіки, можна за різними ознаками: за призначенням, рухливістю, вантажопідйомністю, видом приводів [4–6]. Тому пропонується власна класифікація механічних конструкцій роботів, моделі яких можна відносно легко розробити і впровадити в експлуатацію. У цьому випадку найкращим варіантом класифікації буде поділ на класи за принципом дії роботів та взаємодії з навколишніми предметами. Розроблений класифікатор представлений на рис. 1.1. Для кращого розуміння відмінностей між наведеними на класифікаторі типами механічних частин роботів нижче наведено короткі описи кожного їх виду.

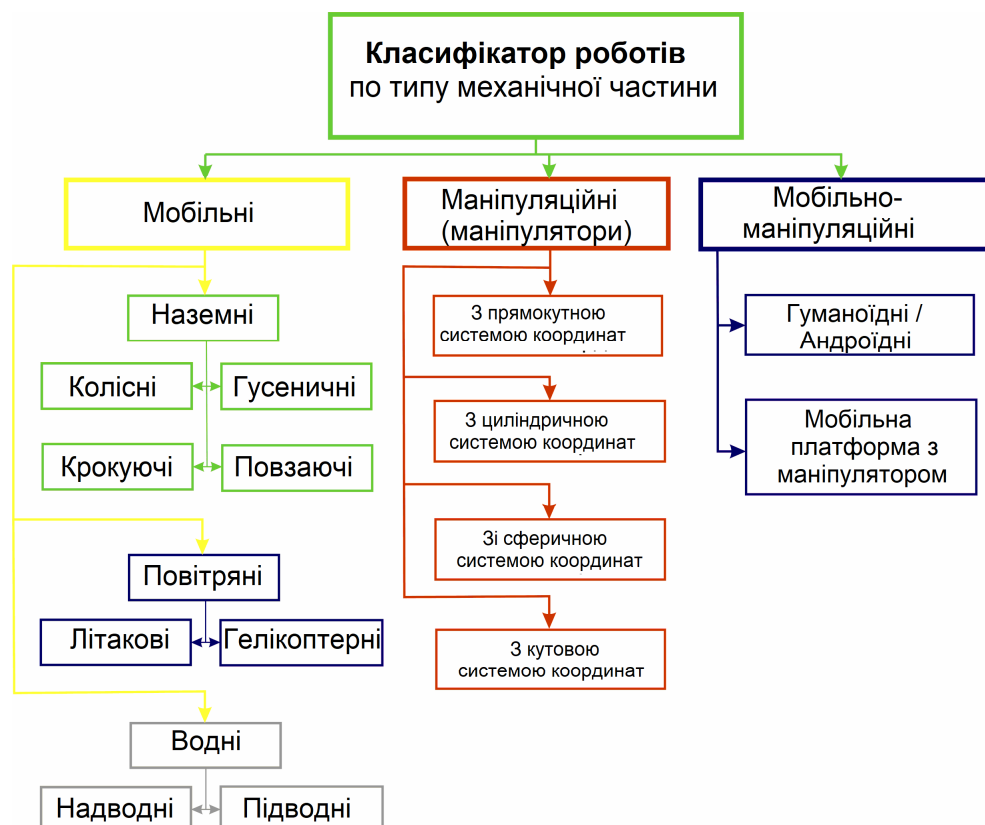
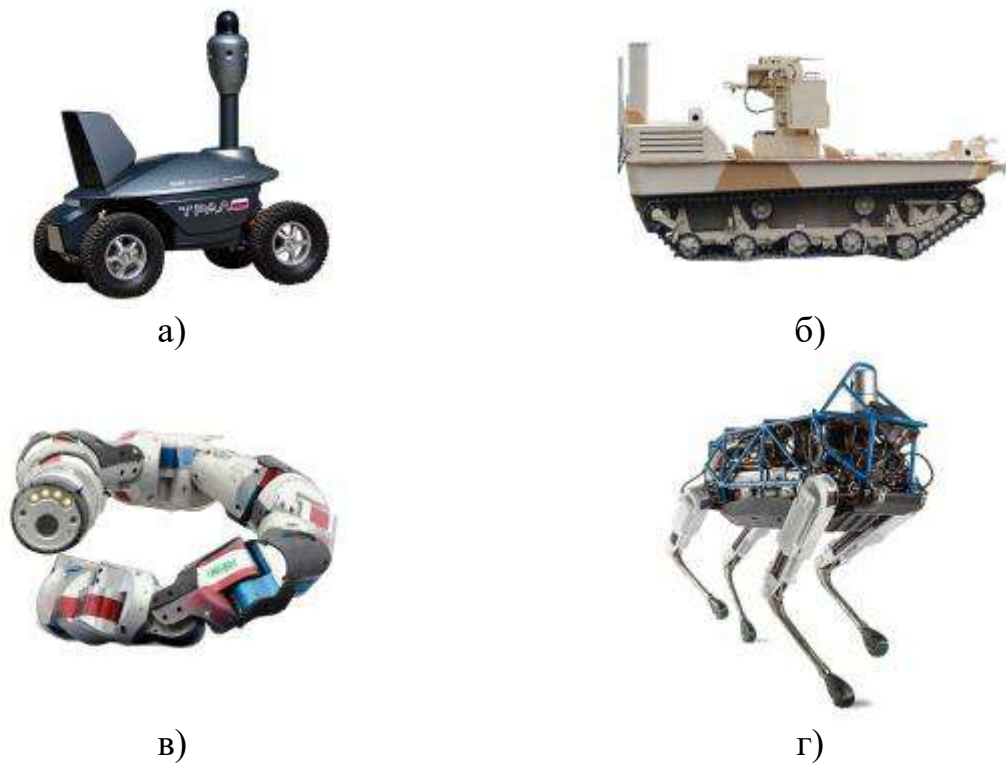


Рисунок 1.1 – Класифікація по типу механічної частини

Мобільні. Це роботи, принцип дії яких нерозривно пов'язаний з переміщенням у просторі, тобто із рухом. Переміщатися і рухатися можуть у різних фізичних середовищах. Можливе створення робототехнічних машин для переміщення по землі чи будь-якій твердій поверхні [7–9]. А можуть створюватись і роботи для польотів у повітряному просторі або плаваючі у водному середовищі [10–12].

Наземні. Такий вид роботів є найпопулярнішим. Це логічно, тому що на суші найбільше роботи, яку треба виконувати. І вона дуже різноманітна. А значить і машини, що її виконують, мають велику кількість різновидів. Вони можуть їздити за допомогою коліс, котитися на гусеницях, крокувати як тварини або люди, або навіть повзати як змії [13–15] (рис. 1.2)



а) колісні; б) гусеничні; в) повзаючі; г) крокоючі
Рисунок 1.2 – Види наземних роботів [16–19]

Повітряні. До цієї категорії відносяться всі літаючі роботи. Літати можуть по-різному: як літак (рис. 1.3, а) чи як гелікоптер (рис. 1.3, б). Роботи гелікоптерного типу називають коптерами (copter у перекладі з англ. – гелікоптер). Вони мають 4 або більше гвинтів, можуть злітати та приземлятись вертикально. Ця здатність робить їх застосування дуже

зручним, оскільки не потрібний великий майданчик для зльоту або посадки. Тому вони найпопулярніші [20–24].



а)



б)

а) літакового типу (БПЛА) [25];

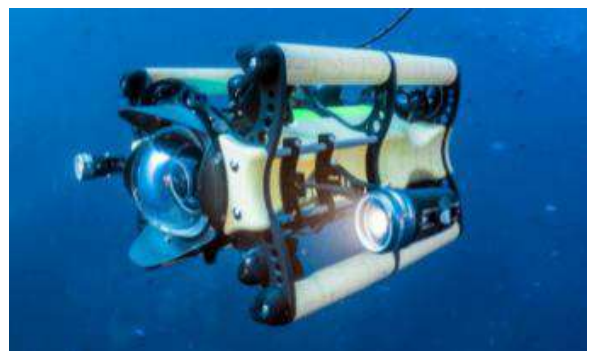
б) гелікоптерного типу (дрони) [26]

Рисунок 1.3 – Повітряні роботи

Водні. Очевидно, що маються на увазі роботи, які призначені для плавання по воді або під водою. Перші представляють кораблі чи човни з автоматичним керуванням (рис. 1.4, а), а другі відносяться до класу підводних човнів чи батискафів (рис. 1.4, б). Дані машини є дуже спеціалізованими і досить рідкісними, порівняно з наземними чи навіть із повітряними роботами, так як покликані вирішувати дуже вузьке коло задач [27–30].



а)



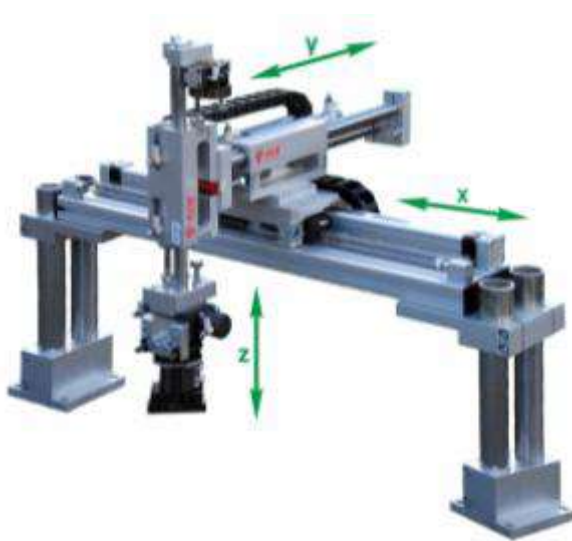
б)

а) надводний патрульний безпілотний катер [31];

б) підводний дистанційно керований апарат [32].

Рисунок 1.4 – Роботи, що працюють у водній стихії

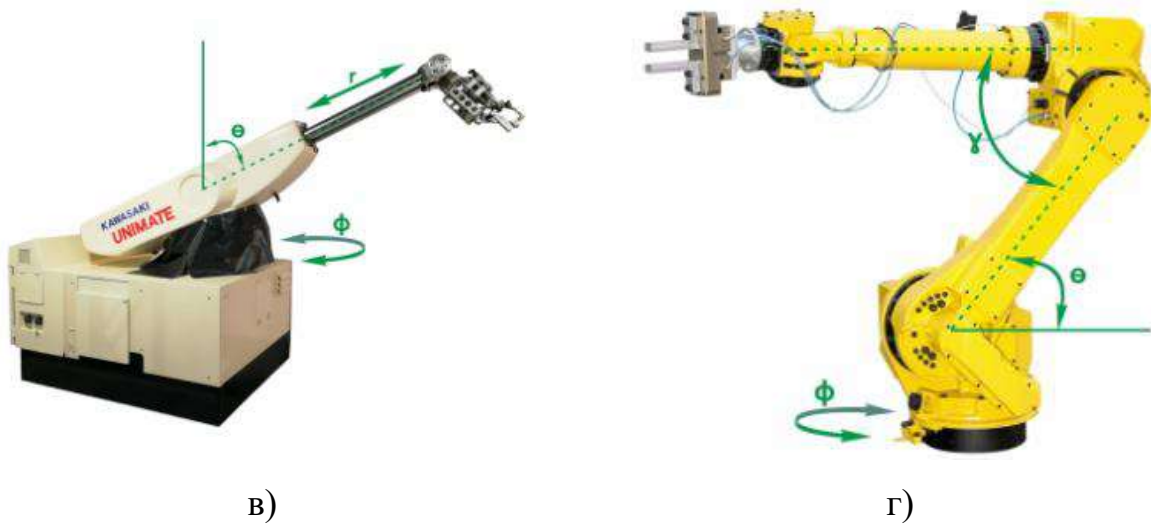
Маніпуляційні. Вперше вони стали використовуватися у промисловості ще у 60-х роках минулого століття. Вони суттєво відрізняються за принципом дії та конструкцією від мобільних роботів. Маніпуляційні роботи призначені для операцій по зміні просторового положення будь-яких предметів, таких як інструменти, деталі чи вузли інших машин. Вони стаціонарні, тобто встановлені в одному місці. Маніпуляції із предметами виконують щодо місця свого закріплення. Маніпуляційні роботи з'явилися і набули широкого поширення саме на промислових виробництвах, оскільки там найпростіше створити умови для безпечної та ефективної роботи такого робота. В результаті за ними закріпилася назва «промисловий робот» [33–35]. На рис. 1.5 показані основні види маніпуляційних роботів, які визначаються системами координат, що використовуються для керування положенням маніпулятора.



а)



б)



а) у прямокутній системі координат [36]; б) у циліндричній [37]; в) у сферичній [38]; г) у кутовій [39].

Рисунок 1.5 – Приклади конструкцій роботів-маніпуляторів, що працюють у різних системах координат

Мобільно-маніпуляційні. Ці роботи являють собою комбінацію маніпуляційних та мобільних роботів. Мобільна платформа пропонує необмежений робочий простір маніпулятора. Додаткові ступені свободи мобільної платформи надають користувачеві більше можливостей. Проте робота такої системи складна через багато ступенів свободи та неструктурованість середовища, в якому вона працює. На даний момент мобільна маніпуляція є предметом серйозної уваги в галузі розробки та досліджень, а мобільні маніпулятори, автономні або дистанційні, використовуються в багатьох галузях, таких як космічні дослідження, військові операції, догляд та медичне обслуговування. Однак у промисловості впровадження мобільних маніпуляторів було обмежено, хоча потреби в інтелектуальній та гнучкій автоматизації присутні. Крім того, необхідні апаратні рішення (мобільні платформи, маніпулятори роботів, зір та інструменти) зараз доступні готовими компонентами для вирішення тих чи інших завдань [40–43]. Приклад мобільно-маніпуляційних роботів наведено рис. 1.6.



а)



б)



в)



г)

а) Mobile manipulator RB-VULCANO;

б) Mobile manipulator XL-GEN;

в) Mobile manipulator FETCH;

г) Mobile manipulator RB-EKEN

Рисунок 1.6 – Мобільно-маніпуляційні роботи

Robotnik Automation [44]

Порівняльні характеристики мобільно-маніпуляційних роботів наведено у таблиці 1.1

Таблиця 1.1 – Порівняльні характеристики мобільно-маніпуляційних роботів Robotnik Automation

Mode N°	Ambient temperature	Size	Weight	Speed	Load capacity	Traction motors	Baterías	Communication	PC
FETCH [45]	0° to +40°C	-	113.3 Kg	1,0 m/s	6 Kg			WiFi 802.11n	Open architecture ROS
XL-GEN [46]	0° to +40°C	720 x 613 x 708 mm	50 Kg	3 m/s	7 DOF: 2,4 kg		LiFePO4 15Ah@24V	WiFi 802.11n	Open architecture ROS Embedded PC with Linux
RB-VULCANO [47]	0° to +45°C	1.606 x 1.096 x 999 mm	850 kg	1 m/s	1.750 Kg	4x1200 W	-	WiFi 802.11n	Open architecture ROS Embedded PC with Linux
RB-EKEN [48]	0° to +40°C	1.204 x 730 x 991 mm	270 kg	2 m/s	300 Kg	4x1,2 Kw	60Ah@48V	WiFi 802.11n	Open architecture ROS Embedded PC with Linux
RB-KAIROS [49]	0° to +50°C	978 x 776 x 1.010 mm	115 kg + UR arm	1,5 m/s	100 kg / 250 k	4 x 500 W	LiFePO4 15/30Ah@48V	WiFi 02.11a / b / g / n / ac Bluetooth 5.1	Open architecture ROS PC with Linux integrated
Jaguar V6 [50]	0° to +40°C			0,3 m/s		4 x 80 W	LiPo 22.2V 20AH	WiFi 802.11n	ull development kit including SDK, data protocol and sample codes, supporting Microsoft® Robotics Studio,
Dr. Robot Scout2 [51]	0° to +40°C	430 x 380 x 540 mm	7 kg	0.75 m/sec	2 x 5DOF Arm		2x 3800mAh	WiFi 802.11n	COUT Pro includes all WiRobot development software components (for MS Windows 2000 and up), MS VB and VC++.

Відповідно до проведеного аналізу, основними завданнями мобільно-маніпуляційного робота є позиціонування робочого органу та відслідковування заданої траєкторії.

Позиціонування або переміщення маніпулятора в задану точку, без необхідності проходження певної траєкторії, вимагає врахування насамперед кінематичної структури маніпулятора [52]. У даному випадку динамічні характеристики, під якими розуміються фізичні сили, що діють на маніпулятор загалом, або його компоненти, не мають істотного впливу на якість виконання технологічної операції. Сучасні загальнопромислові маніпулятори забезпечують точність позиціонування порядку 0,05–0,1 мм [53–55].

Рух заданою траєкторією можна розглядати, як сукупність безлічі малих переміщень між окремими точками [56]. Слід зазначити, що в даному випадку закони зміни швидкості та прискорення мають бути визначені виходячи з вимог дотримання заданої точності позиціонування у будь-якій точці траєкторії.

Відмінною особливістю контурних переміщень є значний вплив динамічних параметрів маніпулятора на якість управління рухом. Насамперед, це проявляється у зміні фізичних параметрів окремих ланок маніпулятора (моменти інерції щодо зчленувань, сили взаємодії окремих ланок) [57].

Залежно від призначення мобільного-маніпуляційного робота, для здійснення переміщень маніпулятора із захватним пристроєм використовують різні види приводів, класифікація яких наведено на рис. 1.7.

У завданнях, які вимагають створення значних постійних зусиль, зокрема для управління робочим органом маніпулятора, можуть бути використані пневматичні (для маніпуляторів з малою вантажопідйомністю) та гідравлічні (з великою вантажопідйомністю) приводи. Насамперед, це відноситься до складальних та транспортно-вантажних операцій, де необхідна надійна фіксація об'єкта. Використання таких приводів дозволить

досягти необхідного результату з найменшими витратами, але передбачає використання зовнішніх джерел енергії [58, 59].

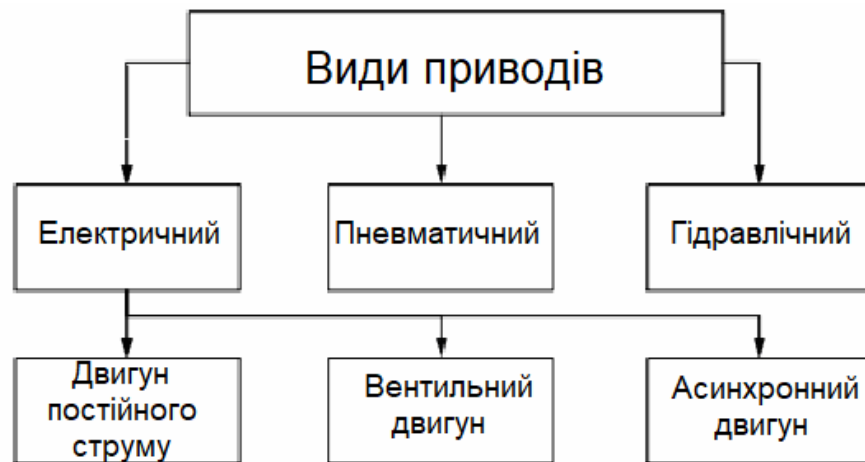


Рисунок 1.7 – Класифікація видів приводів для мобільно-маніпуляційних систем

Більшість існуючих в даний час маніпуляційних роботів використовують для виконання рухів електричний привід. Основними його перевагами щодо пневматичних та гідравлічних приводів є [60–62]:

- високий ККД;
- малі габарити;
- зручність управління;
- широкий діапазон потужностей.

1.2 Механіка маніпуляторів мобільних роботів

До основних завдань механіки маніпуляторів можна віднести [63–65]:

- розробку методів синтезу та аналізу виконавчих механізмів (включаючи механізми приводів);
- програмування руху маніпулятора;
- розрахунок керуючих зусиль та реакцій у кінематичної пари (КП);
- урівноважування механізмів маніпуляторів.

Ці завдання вирішуються на основі загальних методів дослідження структури, геометрії, кінематики та динаміки систем із просторовими рухомими механізмами [66].

Кожне з цих завдань може бути сформульоване як пряме (завдання аналізу) або як зворотне (завдання синтезу). При визначенні функцій положення механізму у прямій задачі знаходять закон зміни абсолютних координат вихідної ланки за заданими законами зміни відносних або абсолютних координат ланок. У зворотній – за заданим законом руху схвату знаходять закони зміни координат ланок, зазвичай лінійних або кутових переміщень у приводах. Вирішення зворотної задачі, або задачі синтезу, складніше, оскільки часто вона має безліч допустимих рішень, з яких необхідно обрати оптимальне. У зворотному завданні кінематики за необхідним законом зміни швидкостей та прискорень вихідної ланки визначаються відповідні закони зміни швидкостей та прискорень у приводах маніпулятора. Зворотне завдання динаміки полягає у визначенні закону зміни керуючих сил та моментів у приводах, що забезпечують заданий закон руху вихідної ланки [67–70].

1.2.1 Кінематичний аналіз механізму мобільного маніпулятора

Перше та основне завдання кінематики – визначення функції положення [71]. Для просторових механізмів найбільш ефективними методами рішення цього завдання – є векторний метод та метод перетворення координат. При вирішенні прямої задачі про положення схвату маніпулятора зазвичай використовують метод перетворення координат [72].

З багатьох методів перетворення координат, які відрізняються один від іншого правилами вибору осей локальних систем координат, для маніпуляторів зазвичай використовується метод Денавіта і Хартенберга [73].

При розв'язанні задачі про положення необхідно: у прямій задачі визначити положення вихідної ланки як функцію переміщень у приводах, у зворотному – задане положення вихідної ланки подати як функцію

переміщень у приводах. Вибір розташування та орієнтації локальних систем координат має забезпечувати виконання цих завдань. При використанні методу Денавіта та Хартенберга вісі координат розташовуються за такими правилами [74]:

- для ланки i вісь z_i спрямовується по осі кінематичної пари, що утворюється ним з ланкою $(i+1)$. Початок координат розміщують у геометричному центрі цієї пари;

- вісь x_i прямує за загальним перпендикуляром до осей z_{i-1} і z_i з напрямком від z_{i-1} до z_i . Якщо осі z_{i-1} і z_i збігаються, то x_i перпендикулярна до них і спрямована довільно. Якщо вони перетинаються в центрі кінематичної пари, то початок координат розташовується в точці перетину, а вісь x_i іде за правилом векторного добутку (найкоротший поворот осі z_i до суміщення з z_{i-1} при спостереженні з кінця x_i повинен відбуватися проти годинникової стрілки);

- вісь y_i спрямовується так, щоб система координат була правою. У прямій задачі необхідно визначити положення захвату маніпулятора та пов'язаної з ним системи координат $Mx_n y_n z_n$ відносно до нерухомої, або базової системи координат $Kx_0 y_0 z_0$. Це здійснюється послідовними переходами із системи координат ланки i в систему координат ланки $i-1$. Відповідно до прийнятого методу кожен перехід включає послідовність чотирьох рухів – двох поворотів і двох паралельних переносів, здійснюваних у зазначеній послідовності (рис. 1.8);

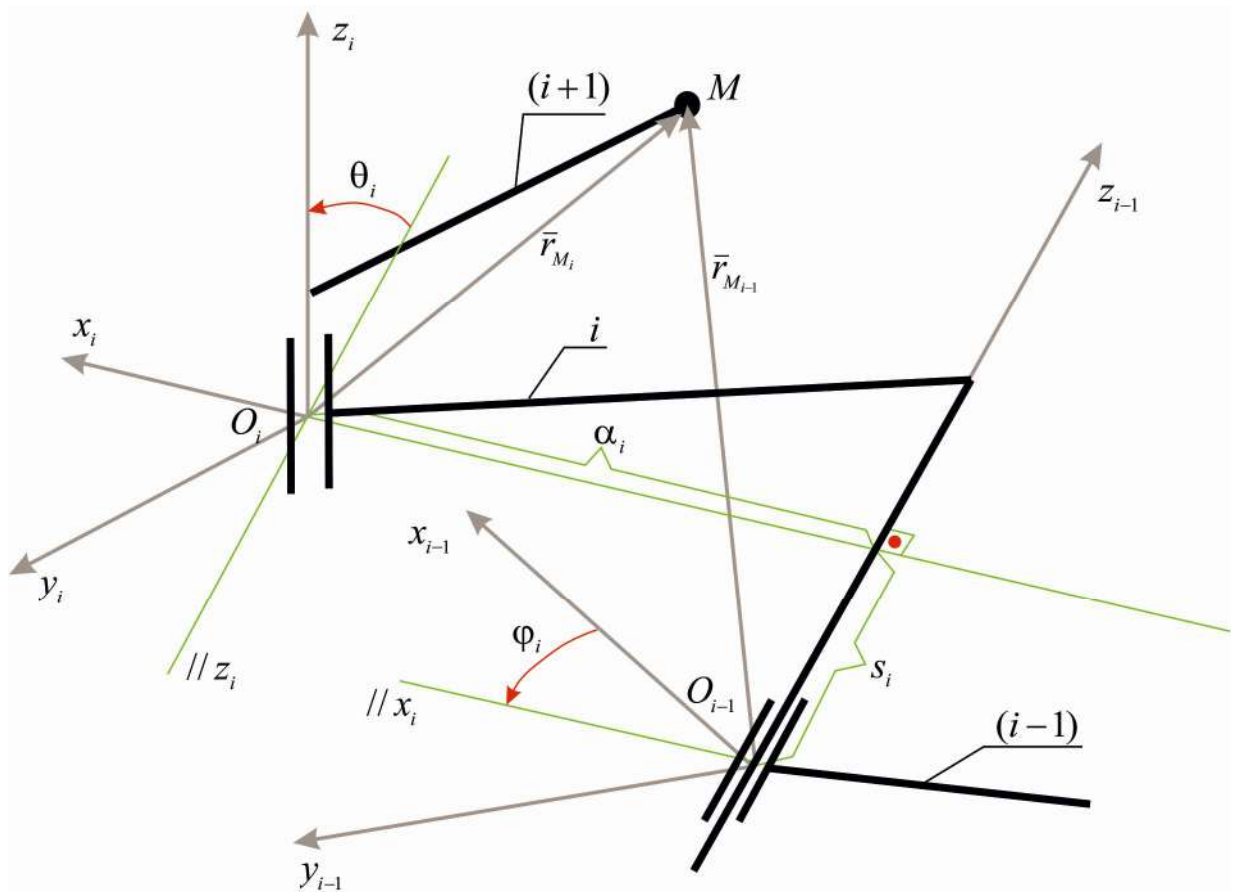


Рисунок 1.8 – Перехід із системи координат ланки i до системи координат ланки $i-1$

- поворот i -ї системи навколо осі x_i на кут $-q_i$ до паралельності осей z_i і z_{i-1} (позитивний напрямок повороту при спостереженні з кінця вектора x_i проти годинникової стрілки);
- перенесення вздовж осі x_j на величину $-a_j$ до суміщення початку системи координат O_j з точкою перетину осей x_j і z_{i-1} (відлік по осі x_j від точки перетину осі x_j та осі z_{i-1});
- перенесення вздовж осі z_{i-1} на величину $-s_j$, після якого початок системи координат O_j виявляється на початку координат O_{j-1} системи $(j-1)$ (відраховується по осі z_{j-1} від початку координат O_{j-1} до точки її перетину з віссю x_j);

– поворот навколо осі z_{j-1} на кут $-j_j$ доти, поки вісь x_j не стане паралельною осі x_{j-1} (позитивний напрямок повороту при спостереженні з кінця вектора z_{j-1} проти годинникової стрілки).

Слід зазначити, що знак кута повороту немає значення, оскільки у матрицях переходу використовуються напрямні косинуси (парні функції). Доцільно розглядати кут, що забезпечує найкоротший поворот осі старої системи j до суміщення (паралельності) з відповідною віссю нової $(j-1)$. Переміщення початку координат визначаються як координати початку старої системи O_j у новій O_{j-1} .

У мобільних робототехнічних маніпуляторах зазвичай використовуються однорухливі кінематичні пари – або обертальні або поступальні [75]. Обидва відносні рухи, як обертальний, так і поступальний, реалізуються в циліндричних парах. Тому за загального представлення механізму використовуються (рис. 2.1) циліндричні пари. Матриці переходу їх системи O_i до системи O_{i-1} можна записати так:

$$M_i = M_i^{\ominus} \cdot M_i^{\alpha} \cdot M_i^S \cdot M_i^{\varphi} \quad (1.1)$$

де: $M_i^{\ominus}$ – матриця повороту навколо осі x_i на кут $-q_i$;

M_i^{α} – матриця перенесення вздовж осі x_i на $-a_i$;

M_i^S – матриця перенесення вздовж осі z_{i-1} на $-s_i$;

M_i^{φ} – матриця повороту осі z_{i-1} на $-s_i$.

У цих матрицях змінні s_i і j_i відповідають відносним переміщенням ланок у кінематичних парах і є узагальненими координатами маніпулятора, що визначають конфігурацію механізму у положенні. Змінні a_i та q_i визначаються конструктивним виконанням ланок мобільного маніпулятора, у процесі руху вони залишаються незмінними.

Положення деякої довільної точки M у системі координат ланки i визначається вектором $\bar{r}_{M_{i-1}}$, а у системі координат ланки $(i-1)$ – вектором $\bar{r}_{M_{i-1}}$. Ці радіуси пов'язані між собою через матрицю перетворення координат M_i наступним рівнянням:

$$\bar{r}_{M_{i-1}} = M_i \cdot \bar{r}_{M_i}, \quad (1.2)$$

де: M_i – матриця переходу з i -ї системи координат в $(i-1)$.

$$M_i = \begin{vmatrix} \cos \varphi_i & -\cos \theta_i \cdot \sin \varphi_i & \sin \varphi_i \cdot \sin \theta_i & \alpha_i \cdot \cos \phi_i \\ \sin \varphi_i & \cos \theta_i \cdot \cos \varphi_i & -\cos \varphi_i \cdot \sin \theta_i & \alpha_i \cdot \sin \phi_i \\ 0 & \sin \theta_i & \cos \theta_i & s_i \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (1.3)$$

Розглянемо шестирухомий маніпулятор у вихідному або початковому положенні (рис. 1.9). За початкове становище приймається таке, у якому всі відносні узагальнені координати дорівнюють нулю [76]. Перехід із системи координат будь-якої i -ї ланки до нерухомої або базової системи записується у вигляді:

$$\bar{r}_{M_0} = M_1 \cdot M_2 \cdot \dots \cdot M_i \cdot \bar{r}_{M_i}, \quad (1.4)$$

чи

$$\bar{r}_{M_0} = T_i \cdot \bar{r}_{M_i}. \quad (1.5)$$

де: $\bar{r}_{M_0} = M_1 \cdot M_2 \cdot \dots \cdot M_i$ – матриця перетворення i -ї системи в координати базової системи координат.

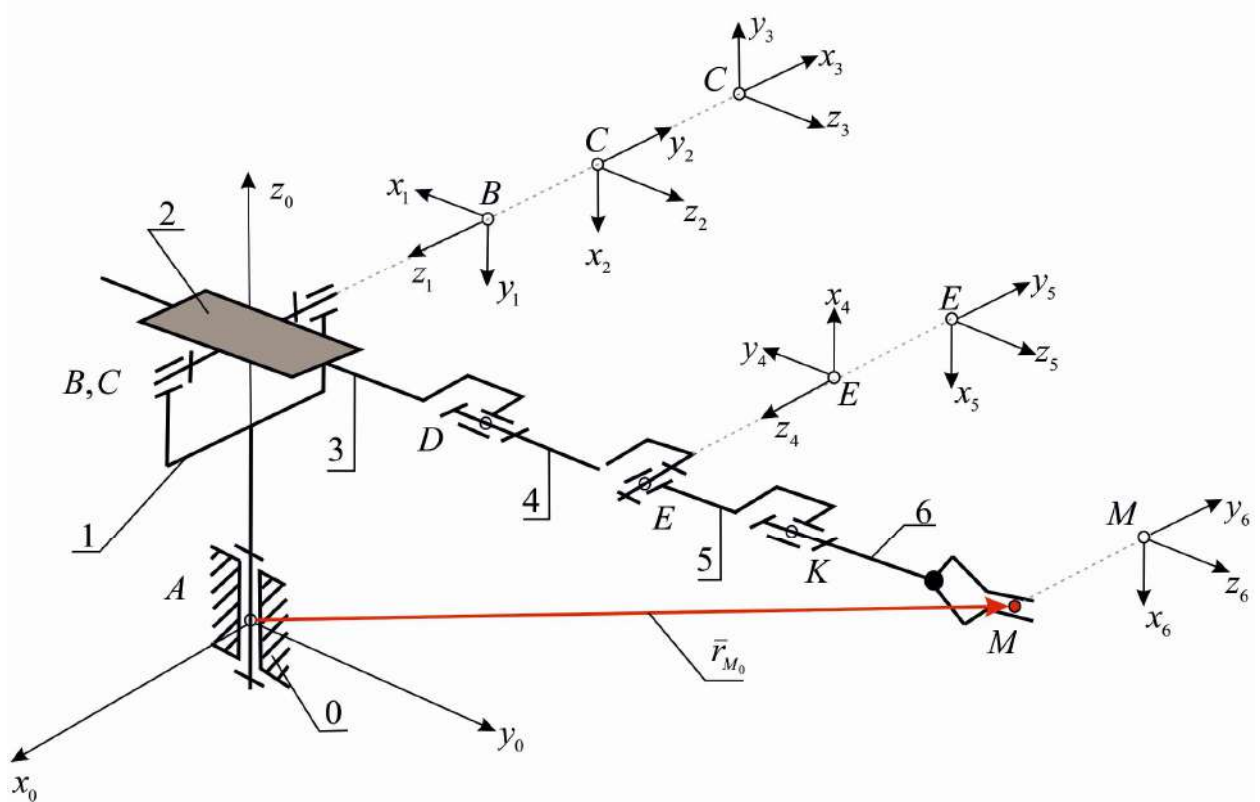


Рисунок 1.9 – Маніпулятор з шістьма напрямками руху у вихідному положенні

Для схеми, зображеної на рис. 1.9, радіус $\bar{r}_{M_6} = 0$, а радіус $\bar{r}_{M_0}$ визначається за формулою:

$$\bar{r}_{M_0} = T_n = T_6 \quad (1.6)$$

Тобто положення вихідної ланки маніпулятора визначається матрицею T_n . Елементи цієї матриці визначають положення центру схвату точки M та орієнтацію його у просторі. Четвертий стовпець визначає декартові координати точки M (проекції вектора $\bar{r}_{M_0}$ на осі координат). Третій стовпець містить напрямні косинуси z_n осі системи координат, пов'язаної зі схватом, або вектор підходу $\bar{A}$, який характеризує напрямок губок схвата (рис. 1.10). Другий стовпець визначає напрямок осі y_n або вектора орієнтації $\bar{O}$, який проходить через центр схвату по осі, перпендикулярній до робочих

поверхонь його губок [77]. У першому стовпці містяться напрямні косинуси осі x_n , або вектора $(\bar{O}, \bar{A})$. Кутом підходу схвата α називається кут між вектором підходу $\bar{A}$ та базовим вектором:

$$R_\alpha = (\bar{O} \times \bar{k}_0) \quad (1.7)$$

де: $\bar{k}_0$ – орт вектора $\bar{z}_0$ нерухомої чи базової системи кардинат. З урахуванням вищенаведеного, матриця T_n може бути представлена в наступному вигляді:

$$T_n = \begin{bmatrix} (\bar{O} \times \bar{A})_x & O_x & A_x & r_n M_x \\ (\bar{O} \times \bar{A})_y & O_y & A_y & r_n M_y \\ (\bar{O} \times \bar{A})_z & O_z & A_z & r_n M_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.8)$$

Орієнтація схвата мобільного маніпулятора представлена рис. 1.10.

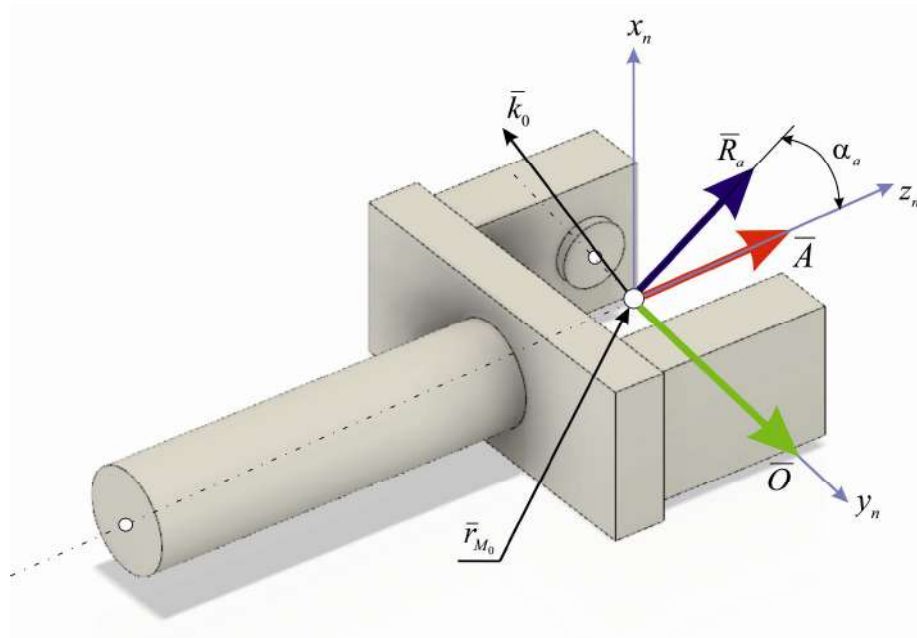


Рисунок 1.10 – Орієнтація схвату маніпулятора

В результаті матричних перетворень отримуємо радіус-вектор точки схвата функції узагальнених координат. Зазвичай, за узагальнені координати

приймають лінійні та кутові переміщення у кінематичних парах, або на вихідних валах приводів маніпулятора [78]. У механізмі з n рухливостями у загальному вигляді функцію положення схвата можна записати так:

$$\bar{r}_{OM} = \bar{T}_n \cdot \bar{A} = \bar{\Pi}(q_1, q_2, \dots, q_n) \quad (1.9)$$

де: $q_1, q_2, \dots, q_n$ – узагальнені координати маніпулятора.

При кінематичному аналізі маніпулятора у прямій задачі необхідно визначити лінійні та кутові швидкості та прискорення схвата, при заданих кутових та лінійних узагальнених швидкостях та прискореннях (зазвичай відносних швидкостях та прискореннях у кінематичних парах механізму). У зворотній задачі, за заданим законом, зміни швидкостей і прискорень схвата визначаються законами зміни швидкостей і прискорень у КП чи вихідних ланках приводів [79]. Вирішення прямої задачі кінематики для точки M схвату можна отримати, продиференціювавши четвертий стовпець матриці T_n за часом:

$$r_{nM} = \begin{vmatrix} r_{nM_x} \\ r_{nM_y} \\ r_{nM_z} \\ 1 \end{vmatrix}; \quad V_{nM} = \frac{dr_{nM}}{dt} = \begin{vmatrix} V_{nM_x} \\ V_{nM_y} \\ V_{nM_z} \\ 1 \end{vmatrix}; \quad a_{nM} = \frac{d^2 r_{nM}}{dt^2} = \begin{vmatrix} a_{nM_x} \\ a_{nM_y} \\ a_{nM_z} \\ 1 \end{vmatrix}. \quad (1.10)$$

Кутову швидкість та кутове прискорення схвата можна визначити векторним підсумовуванням відносних кутових швидкостей у обертальних КП механізму. Оскільки вектори кутових швидкостей при цьому виборі орієнтації осей координат збігаються з віссю z , то кутова швидкість схвата буде визначатись як:

$$\bar{\omega}_n = \sum_{i=1}^n \bar{k}_{i-1} \cdot \omega_i \quad (1.11)$$

де: $\bar{k}_{i-1}$ – орт осі z системи координат, розташованої в центрі КП, що з'єднує ланку i та ланку $i-1$;

m – число обертальних КП у механізмі.

Диференціюючи цей вираз у часі, отримаємо формулу визначення кутового прискорення схвата:

$$\bar{\varepsilon}_n = \sum_{i=1}^m \bar{k}_{i-1} \cdot \varepsilon_i + \sum_{j=2}^m \left[\left(\sum_{k=l-1}^{m-1} \bar{k}_{k-1} \cdot \omega_k \right) \times (\bar{k}_{j-1} \cdot \omega_j) \right]. \quad (1.12)$$

1.2.2 Силевий розрахунок мобільного маніпулятора

З великої різноманітності завдань динаміки маніпуляторів розглянемо дві: силевий розрахунок та розрахунок швидкодії ПР [80]. При силовому розрахунку маніпуляторів вирішуються завдання: визначення зовнішніх силових керуючих впливів, що забезпечують необхідний закон руху механізму; розрахунок реакцій у кінематичних парах. Першу частину часто називають завданням синтезу керування. При силовому розрахунку зазвичай застосовується метод кінетостатики, що ґрунтується на принципі Д'Аламбера [81]. За цим методом до зовнішніх сил і моментів, доданих до ланок механізму, додаються розрахункові сили інерції, які забезпечують силову врівноваженість системи та дозволяють розглядати рухливу систему у квазістатичній рівновазі, тобто як умовно нерухому. Силевий розрахунок виконується при заданому корисному навантаженні F_n , відомих законів руху ланок $\bar{a}_{si}$ та $\bar{\varepsilon}_i$, інерційних характеристиках ланок – масах ланок (m_i) та їх моментів інерції I_{si} . Виходячи з цих даних можна виділити головний вектор для кожної з ланок механізму:

$$\bar{F}_{ui} = -m \cdot \bar{a}_{si}. \quad (1.13)$$

Основні моменти для кожної з ланок механізму:

$$\bar{M}_{ui} = -I_{si} \cdot \bar{\varepsilon}_i. \quad (1.14)$$

Для відкритого кінематичного ланцюга рішення починаємо з вихідної ланки – схвата [82]. Для зручності розуміння силового розрахунку схвату, представимо маніпулятор у такому вигляді, як представлено рис. 1.11.

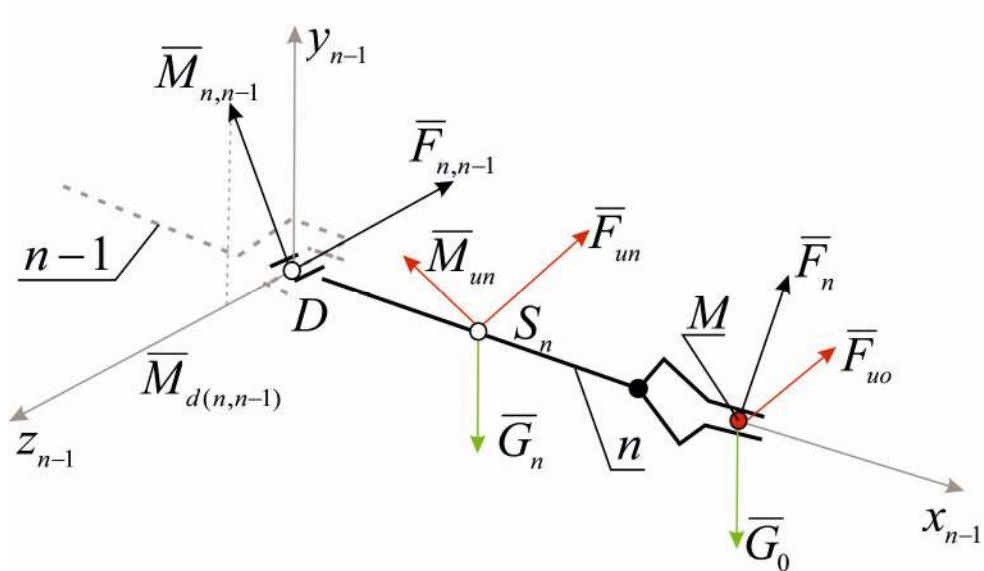


Рисунок 1.11 – Схема силового розрахунку маніпулятора

Відкинуті зв'язки ланки n із ланкою $n-1$ та вихідним валом приводу ланки n замінюємо реакціями $\bar{M}_{n,n-1}$ та $\bar{F}_{n,n-1}$ і складаємо кінетостатичні векторні рівняння рівноваги сил та моментів для ланки n (за рис. 1.11):

$$\bar{G}_0 + \bar{G}_n + \bar{F}_{un} + \bar{F}_{uo} + \bar{F}_{n,n-1} + \bar{F}_n = 0, \quad (1.15)$$

$$\bar{M}(\bar{G}_n) + \bar{M}(\bar{G}_0) + \bar{M}(\bar{F}_{un}) + \bar{M}(\bar{F}_{uo}) + \bar{M}_{un} + \bar{M}_{n,n-1} + \bar{M}(\bar{F}_n) = 0 \quad (1.16)$$

де: $\bar{M}_{n,n-1}$ – вектор моменту в кінематичній парі (проекція це вектор на осі z , є рушійним моментом приводу в КП, тобто $\bar{M}_{z(n,n-1)} = \bar{M}_{d(n,n-1)}$).

Проеціюючи векторні рівняння на осі координат, отримаємо систему з шести рівнянь алгебри, звідки визначимо шість невідомих:

$$F_{x(n,n-1)}, F_{y(n,n-1)}, F_{z(n,n-1)}, M_{x(n,n-1)}, M_{y(n,n-1)}, M_{z(n,n-1)} = M_{d(n,n-1)} \quad (1.17)$$

Далі розглядається рівновага ланки $n-1$. При цьому в місці його приєднання до ланки n прикладаються реакції з боку ланки n , що дорівнюють за величиною і протилежні у напрямку реакцій, визначених на попередньому етапі розрахунку. Так, послідовно, складаються рівняння силової рівноваги всіх n ланок механізму. З рішення отриманої системи $6n$ рівнянь визначаються реакції у кінематичних парах, рушійні сили та моменти [83].

$$F_{x(n-1,n)}, F_{y(n-1,n)}, F_{z(n-1,n)}, M_{x(n-1,n)}, M_{y(n-1,n)}, M_{z(n-1,n)} = M_{d(n-1,n)} \quad (1.18)$$

1.2.3 Врівноважування мобільних маніпуляторів

У більшості кінематичних схем маніпуляторів приводи сприймають статичні навантаження від ваги ланок. Це вимагає значного збільшення потужностей двигунів приводів та моментів гальмівних пристроїв [84]. Для боротьби з цим використовують три методи:

- використовують кінематичні схеми маніпуляторів, в яких сили ваги ланок сприймаються підшипниками кінематичних пар. На потужність приводів і гальмівних пристроїв за такого рішення сили ваги впливають лише через сили тертя у парах. Як приклад можна навести кінематичну схему робота SCARA (рис. 1.12). Недоліком цього методу є великі осьові навантаження в підшипниках [85].

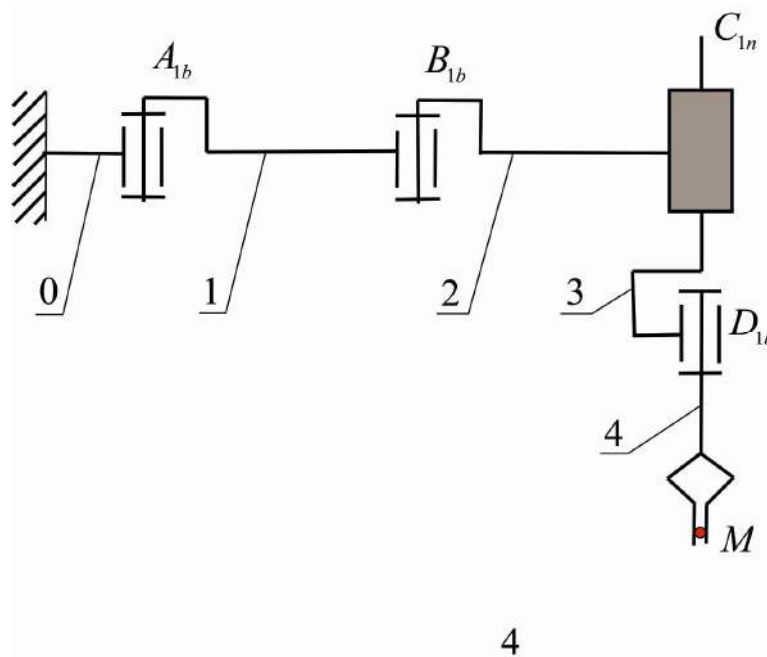


Рисунок 1.12 – Кінематична схема робота SCARA

– врівноваження ланок маніпулятора за допомогою коригування їх маси. При цьому центр мас ланки за допомогою коригувальних мас зміщується до центру кінематичної пари (рис. 1.13). Недоліком цього є значне збільшення маси маніпулятора і моментів інерції його ланок [86].

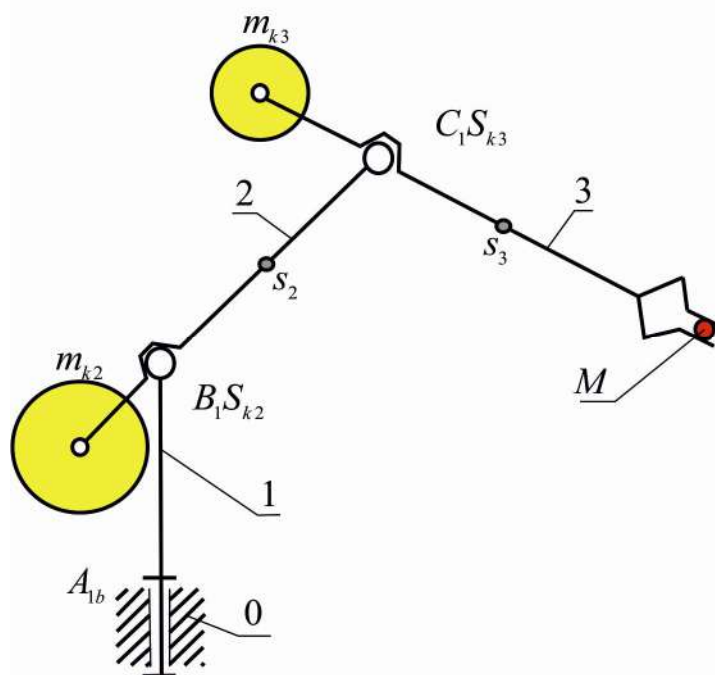


Рисунок 1.13 – Врівноваження ланок противагами

де μ – кількість ступенів свободи того простору, в межах якого працює механізм ($\mu=6$ необхідно для просторового руху);

N – число рухливих ланок механізму;

p – число кінематичних пар механізму;

i – рухливість пари.

Для переважної більшості подібних просторових механізмів число внутрішніх входів n_{cel} (виконавчих циліндрів) дорівнює ступеню рухливості W , це дозволяє визначити максимальну кількість одночасно керуючих впливів на маніпулятор [89, 90]. Для маніпулятора, що розробляється, ступінь рухливості щодо шасі і відповідно число незалежних узагальнених координат $q = n_{cal} = W = 4$, ця рівність є умовою нормальності механізму представленого на рис. 1.15

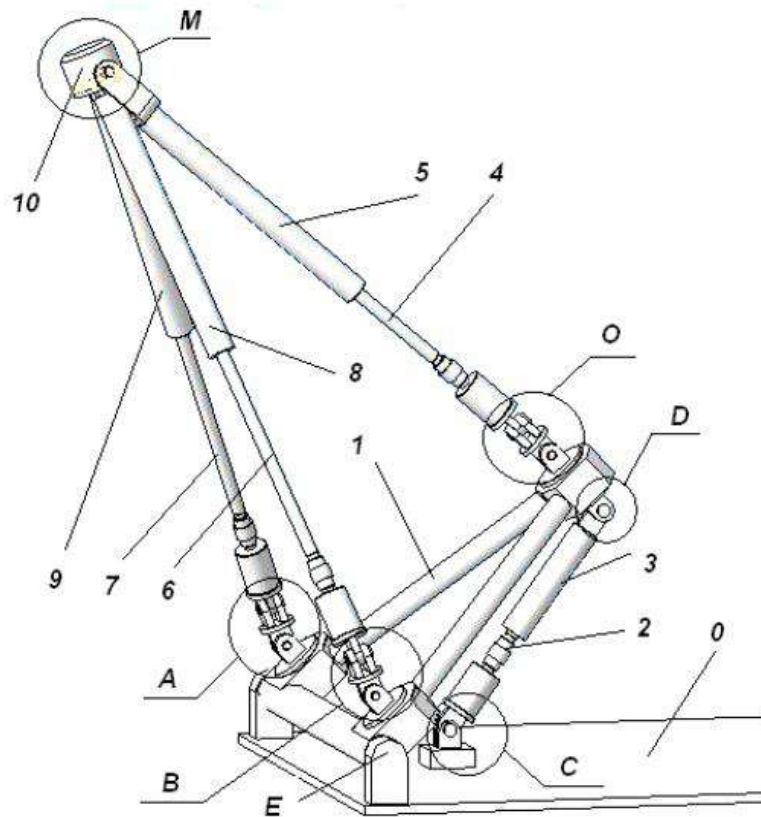


Рисунок 1.15 – Маніпулятор трипод [91]

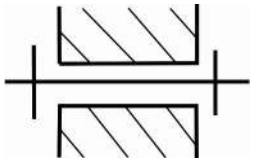
Для синтезу оптимальної за структурою маніпулятора визначимо з (1.19), за цільовою функцією $W = 4$. Отже, для розроблюваного типу

маніпулятора, число рухомих ланок не може бути менше 10. Визначимо рухомі ланки у відповідності до рис. 1.15: схват, поворотна платформа 1 і чотири циліндри, що складаються з поступальних пар штокциліндрів: 2-3, 4-5, 6-7, 8-9), з урахуванням можливих проміжних ланок у складових шарнірів приймається обмеження $10 \leq N \leq 15$.

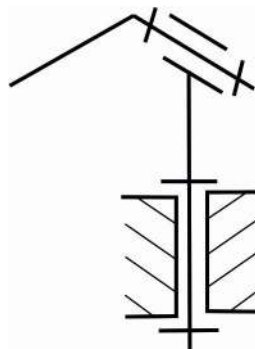
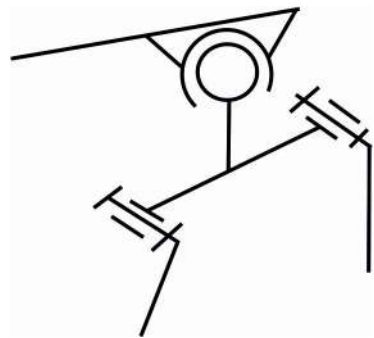
Мінімальна кількість однорухливих поступальних пар нижчого класу дорівнює семи – кінематичні пари платформа-шасі (E), шток-платформа (D), циліндр-шасі (C) і чотири пари шток-циліндрів. З урахуванням застосування в механізмі тільки однорухливих пар 5 класу їх максимальна кількість буде 17, тоді $7 \leq p_5 \leq 16$. Обмеження за дворуховими та трируховими кінематичними парами нижчого класу запишуться як: $0 \leq p_4 \leq 5$ і $0 \leq p_3 \leq 6$. Сумарна кількість кінематичних пар для даної схеми маніпулятора не може бути меншою за 12, тому $p_5 + p_4 + p_3 \geq 12$.

Таким чином, вирішуючи оптимізаційну задачу можна визначити оптимальну структуру маніпулятора з необхідною мірою рухливості, позбавленого надлишкових зв'язків і не маючого місцевих рухливостей [92]. Найбільш раціональні варіанти виконання кінематичних пар у зчленуваннях маніпулятора наведено у таблиці 1.2.

Таблиця 1.2 – Варіанти виконання кінематичних пар у зчленуваннях маніпулятора [93]

№	Опис зчленування ланок	Схема
1	2	3
1	Однорухливі C, D, E	

Продовження таблиці 1.2

1	2	3
2	Дворухливі A, B, O	
3	П'ятирухливі M	

Геометричні розміри ланок визначаються з умови реалізації необхідного руху, зокрема необхідної зони дії, виходячи з геометричних параметрів мобільної платформи [94–96]. Основними геометричними обмеженнями синтезу маніпулятора, на основі просторового механізму, визначимо:

- кут сектора зони в горизональній площині $\Theta \geq 90^\circ$;
- сторона основи піраміди виконавчого механізму $a < 500$ мм;
- нижня точка зони дії схвату щодо поверхні $Z_{\min} = 50$ мм;
- максимальна висування схвату від передньої корми робота

$$L_{\max} \geq 300 \text{ мм.}$$

Схема маніпулятора, з урахуванням просторового виконавчого механізму, на базі мобільного гусеничного робота представлена рис. 1.16.

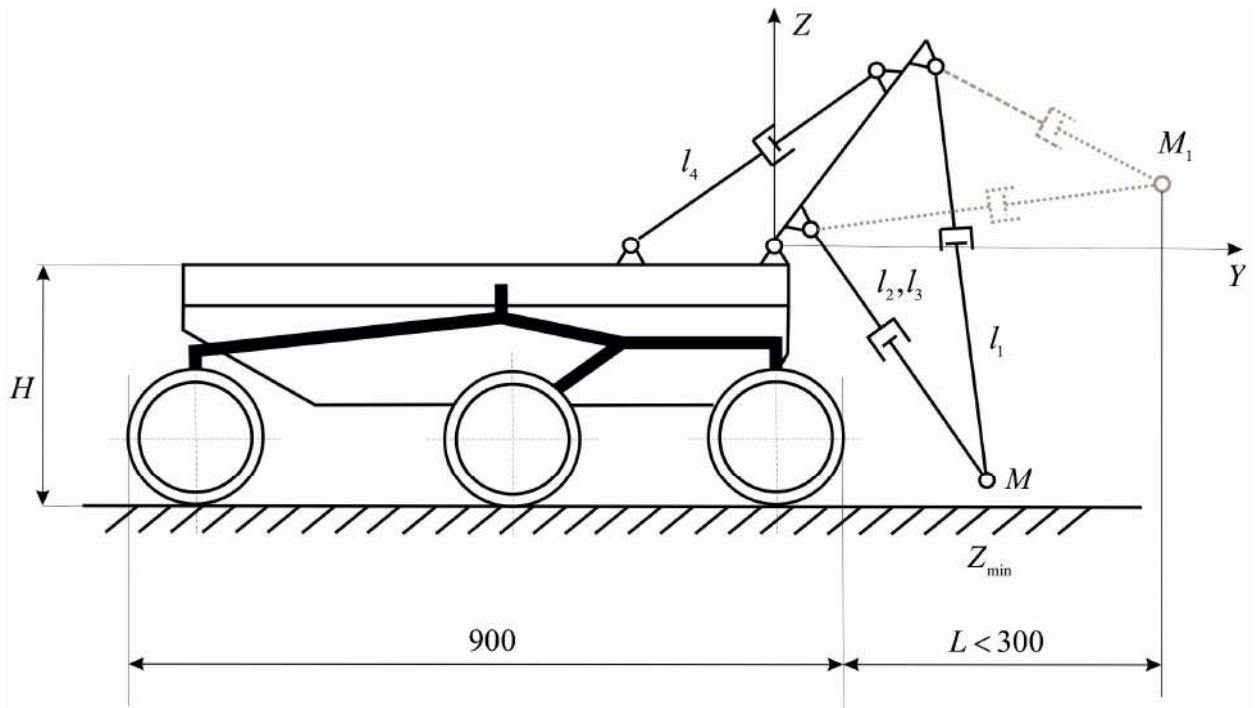


Рисунок 1.16 – Схема маніпулятора на основі просторового виконавчого механізму мобільного гусеничного робота

Найбільше значення кута сектора Θ буде у випадку, коли ΔAMB паралельний площині XOY (рис. 1.17), і коли довжина циліндра l_2 буде мінімальною, а l_3 максимальною у крайньому лівому положенні, і навпаки l_2 буде максимальною, а l_3 мінімальною у крайньому правому положенні. Так як механізм симетричний щодо осі OY , то досить розглянути його рух при постійній мінімальній довжині циліндра l_2 та змінити тільки l_3 .

Взявши вище зазначені припущення, що $\Delta AMB \parallel XOY$ і $l_2 = \text{const}$ дозволить отримати плоску заміщувальну схему з циліндром l_3 , що коливається, і коромислом l_2 . Тоді і лише тоді, при переході з одного крайнього положення AM_1B в інше AM_2B довжина циліндра l_3 змінюється від $l_{3\min}$ до $l_{3\max}$, повертаючи циліндр l_2 на кут α . Подовження циліндра позначимо за $h = l_{3\max} - l_{3\min}$ відношення довжини циліндра до ходу штока є коефіцієнт подовження циліндра $k = l_3 / h$. Для гідроциліндрів мобільного

робота $k = 1, 4 \dots 3$, припускаючи використовувати електроциліндри (актуатори), надалі приймемо діапазон змін $k = 1 \dots 2$.

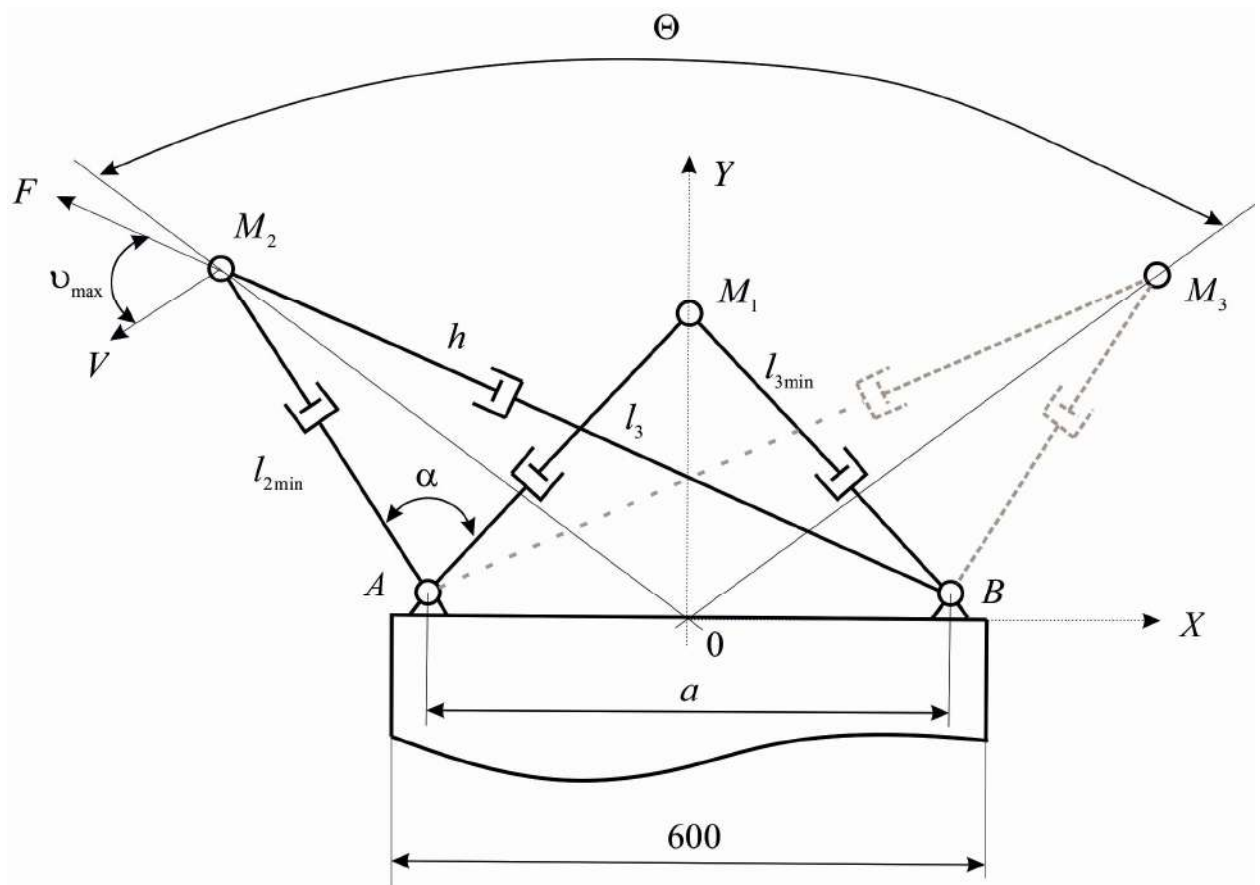


Рисунок 1.17 – Розрахункова схема маніпулятора до кінематичного синтезу (вид зверху)

Слід враховувати, що працездатність механізму та його ККД багато в чому залежить від кута тиску ν – кута між віссю циліндра та вектором швидкості точки докладання сили. Допустимий кут тиску в важільних механізмах, як правило, не повинен перевищувати $[\nu] = 60^\circ$. Оптимальна за габаритами схема механізму буде за умови $\nu_{\max} = [\nu]$, а з іншого боку, чим менше $[\nu]$, то менше тертя в шарнірах.

Відповідно до схеми (рис. 1.17) отримаємо рівняння:

$$(k+1)^2 \cdot h^2 - 4l_{2\min} \cdot \sin\left(\frac{\alpha}{2}\right) \cdot (k-1) \cdot \cos\left([\nu] - \frac{\alpha}{2}\right)h + 4l_{2\min}^2 \cdot \sin^2\left(\frac{\alpha}{2}\right) - l_{3\min}^2 = 0 \quad (1.20)$$

рішенням якого є

$$h = -\frac{b}{2} + \sqrt{\frac{b^2}{4} - c} \quad (1.21)$$

$$b = -4l_{2\min}(k+1) \cdot \sin\left(\frac{\alpha}{2}\right) \cdot \cos\left([\nu] - \frac{\alpha}{2}\right) / (2k+1) \quad (1.22)$$

$$c = 4l_{2\min}^2 \cdot \sin^2\left(\frac{\alpha}{2}\right) / (2k+1) \quad (1.23)$$

Після визначення рішення рівняння (1.20) знаходимо максимальне подовження циліндра l_3 та довжину стійки a :

$$l_{3\max} = k \cdot h \quad (1.24)$$

$$a = \sqrt{l_{2\min}^2 + (l_{3\min} + h)^2 - 2l_{2\min}(l_{3\min} + h) \cdot \sin[\nu]} \quad (1.25)$$

Кут повороту циліндра α залежить від кута сектора зони дії схвату Θ як:

$$\alpha = \frac{\Theta + [\nu] - \pi / 6}{2} \quad (1.26)$$

Залежності кута зони дії Θ від різних граничних кутів тиску $[\nu]$ наведено на рис. 1.18.

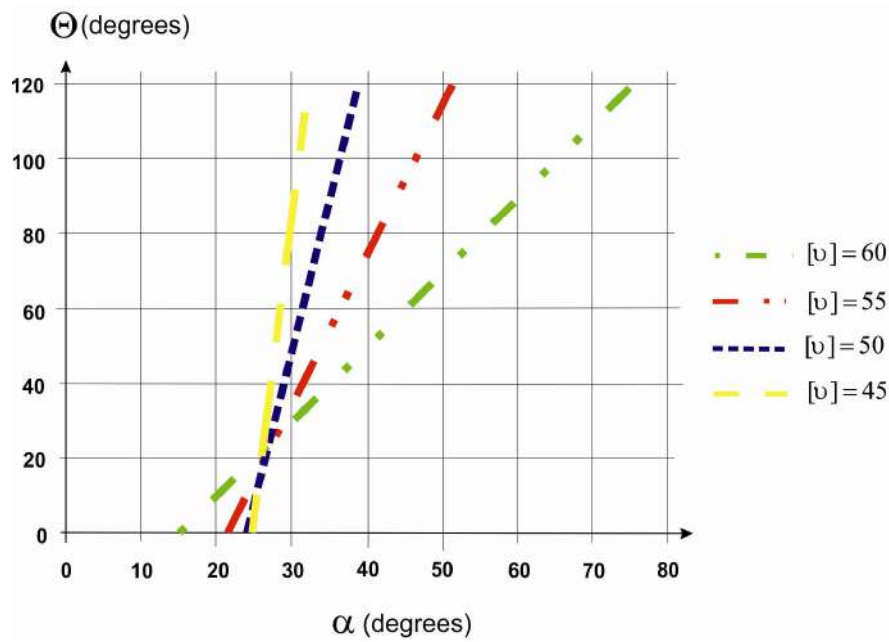


Рисунок 1.18 – Залежність кута зони дії Θ від кута повороту циліндрів α при різних граничних кутах тиску $[\nu]$

Визначивши подовження h циліндра l_3 та міжосьову відстань a , приймаємо однаковими розміри всіх циліндрів, та однаковими міжосьові відстані основи піраміди, з метою симетричності просторового механізму маніпулятора. Грунтуючись на симетричності маніпулятора та припускаючи, що висота кріплення нижніх шарнірів основи піраміди просторового механізму маніпулятора від поверхні дорівнює висоті платформи робота, отримаємо такі умови:

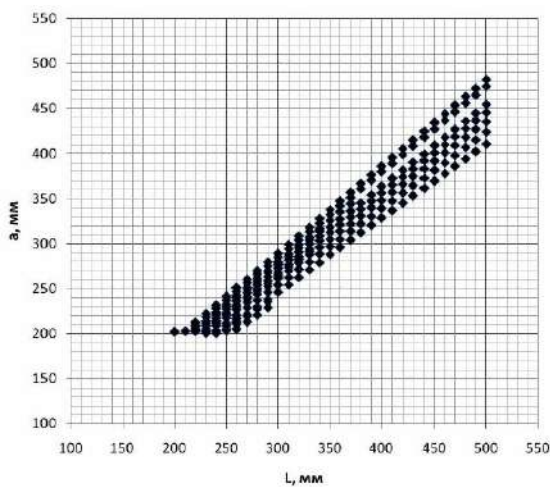
$$\sqrt{l_2^2 - a^2 / 4} > (H_{pl} - z_{min}) \quad (1.27)$$

з якого визначаються мінімальні значення довжин циліндрів l_2 та l_3 . Параметри циліндра $l_4(k_1 \& h_1)$ визначаються залежно від аналогічних рівнянь (1.20)–(1.25). З усіх раніше знайдених параметрів визначається максимальний виліт схвату від кромки платформи робота:

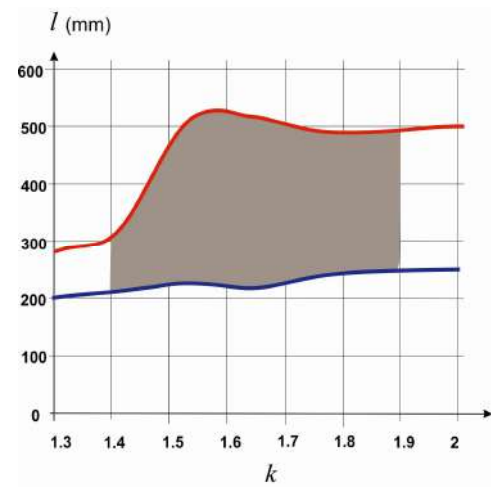
$$L_{max} = \sqrt{l_{1max}^2 - 3a^2 / 16} \quad (1.28)$$

Розрахунок параметрів маніпулятора за заданими вище обмеженнями дозволило отримати залежності довжин циліндрів l_i і міжосьових відстаней a від коефіцієнта подовження k наведено на рис. 1.19.

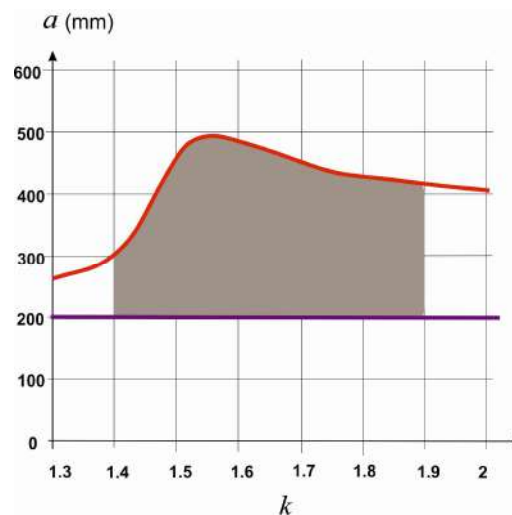
Область допустимих значень довжини основи піраміди a від довжини циліндрів L , наведені на рис. 1.19, а, область допустимих значень l , залежно від k , за умови забезпечення необхідної зони дії наведені 1.19, б, область допустимих значень a , залежно від k , за умови забезпечення необхідної зони дії (рис. 1.19, в).



а)



б)



в)

Рисунок 1.19 – Область допустимих значень параметрів з умовами для забезпечення необхідної зони дії

Основні оптимальні параметри маніпулятора мобільного робота при $k = 1,7$ забезпечення кута сектора зони дії $\Theta = 1000$, що задовольняють усім раніше прийнятим обмеженням наведені в таблиці 1.3.

Таблиця 1.3 – Оптимальні результати синтезу

l_i (мм)	330	340	350	360	370	380	390	400
a (мм)	287.4	296.1	304.9	313.3	322.3	331.1	339.9	348.4
$L_{\max}$ (мм)	303.7	312.9	322.2	331.3	340.4	349.6	358.9	368.3

При куті сектора зони дії $\Theta = 1000$ і $k = 1,7$ міжосьова відстань між точками кріплення циліндрів на платформі і максимальний виліт схвату від краю робота визначаються як $a = 0,876 \cdot l_i$, а $L_{\max} = 0,92 \cdot l_i$. Однак, отримані в результаті синтезу параметри маніпулятора можуть відрізнятися від реальної конструкції, оскільки залежно від вантажопідйомності маніпулятора можливе збільшення діаметра виконавчих циліндрів, що спричинить невелике зменшення зони дії.

1.3.1 Кінематика маніпулятора-трипода

Особливість кінематичного аналізу самохідного маніпулятора впливає з особливостей самого механізму, в якому всі рухомі ланки є провідними та мають змінну довжину [97–99]. З цієї причини достатньо виявити переміщення єдиної «визначальної» точки, до якої прикріплені штоки трьох циліндрів (актуаторів) та схват.

Теоретична зона можливих переміщень точки M залежить від співвідношення довжин циліндрів L_1, L_2, L_3, L_4 , від взаєморозташування їх точок кріплення і від величини переміщення самого шасі.

Для визначення траєкторії переміщення точки M було прийнято припущення про незалежність трьох рухів, а саме:

I-й рух – за рахунок зміни довжин циліндрів трипода;

II-й рух – за рахунок зміни довжини циліндра L_4 ;

III-й рух – за рахунок переміщення самого шасі у поздовжньому напрямку.

Тоді траєкторія точки M (I-й рух) визначається сумою цих рухів:

$$\begin{cases} X_M = X_{Mi} + X_{Mj} + X_{Mn}; \\ Y_M = Y_{Mi} + Y_{Mj} + Y_{Mn}; \\ Z_M = Z_{Mi} + Z_{Mj} + Z_{Mn}. \end{cases} \quad (1.29)$$

де X_M, Y_M, Z_M – координати точки M в результуючому русі;

X_{Mi}, Y_{Mj}, Z_{Mn} – координати точки M в i, j, n -ому рухах.

Для складання рівнянь зв'язку між координатами точки M , довжинами циліндрів та взаєморозташуванням їх точок кріплення була використана відома залежність між координатами двох точок у просторі та відстанню між ними:

$$\begin{cases} X_{Mi}^2 + Y_{Mj}^2 + Z_{Mn}^2 = L_1^2; \\ (X_{Mi} - a)^2 + (Y_{Mj} - b)^2 + (Z_{Mn} - c)^2 = L_2^2; \\ (X_{Mi} + a)^2 + (Y_{Mj} - b)^2 + (Z_{Mn} + c)^2 = L_3^2. \end{cases} \quad (1.30)$$

де a, b, c – координати точок кріплення циліндрів;

L_1, L_2, L_3 – поточні значення довжин циліндрів 1, 2 та 3 відповідно.

Ці рівняння стосуються схеми, представленої рис. 1.20, і мають такий вигляд для просторового переміщення точки M від зміни $L_1 + L_2 + L_3$

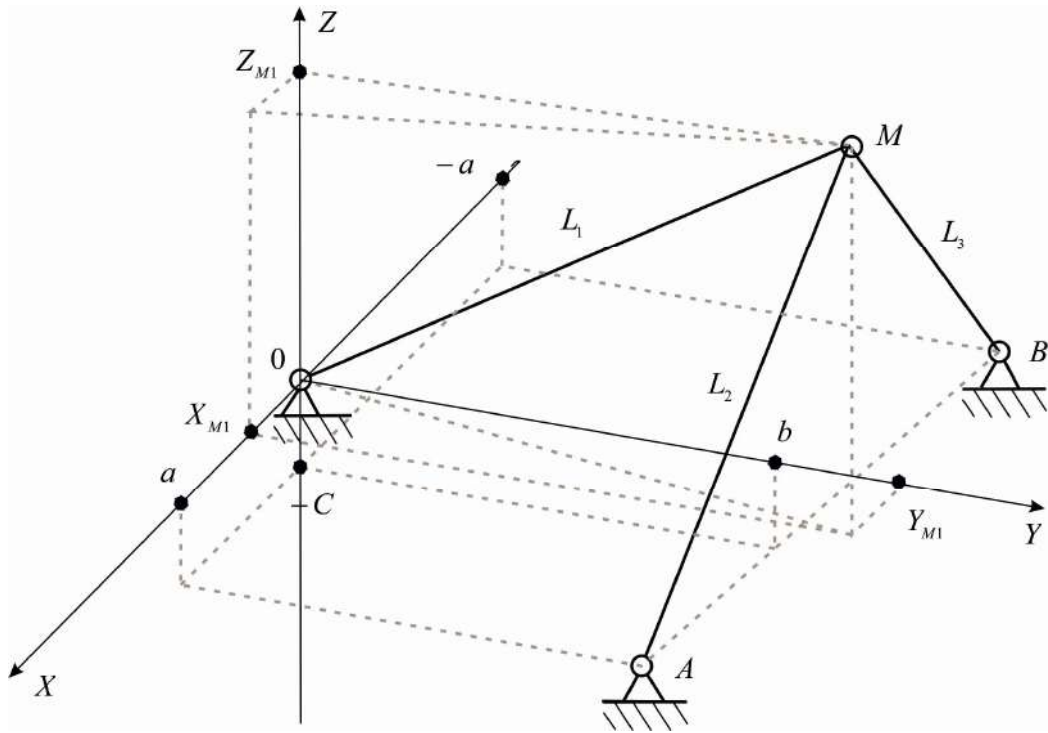


Рисунок 1.20 – Приклад визначення положення точки M у просторі O'

Вирішивши систему (1.30) отримуємо залежності координат точки $M(X_{Mi}, Y_{Mj}, Z_{Mn})$ від довжин циліндрів L_1, L_2, L_3 та координат точок кріплення циліндрів у вигляді системи:

$$\begin{cases} X_{Mi} = \frac{L_3^2 - L_2^2}{4a}; \\ Y_{Mj} = \frac{-abX + c\sqrt{Y}}{4a(b^2 + c^2)}; \\ Z_{Mn} = \frac{acX + b\sqrt{Y}}{4a(b^2 + c^2)}. \end{cases} \quad (1.31)$$

де $X = L_2^2 + L_3^2 - 2L_1^2 - 2(a^2 + b^2 + c^2),$
 $Y = (b^2 + c^2)[16aL_1^2 - (L_3^2 - L_2^2)^2] - a^2 X^2.$

Приклад подання плаского обертання точки M навколо точки C за рахунок зміни довжини циліндра L_4 представлений на рис. 1.21.

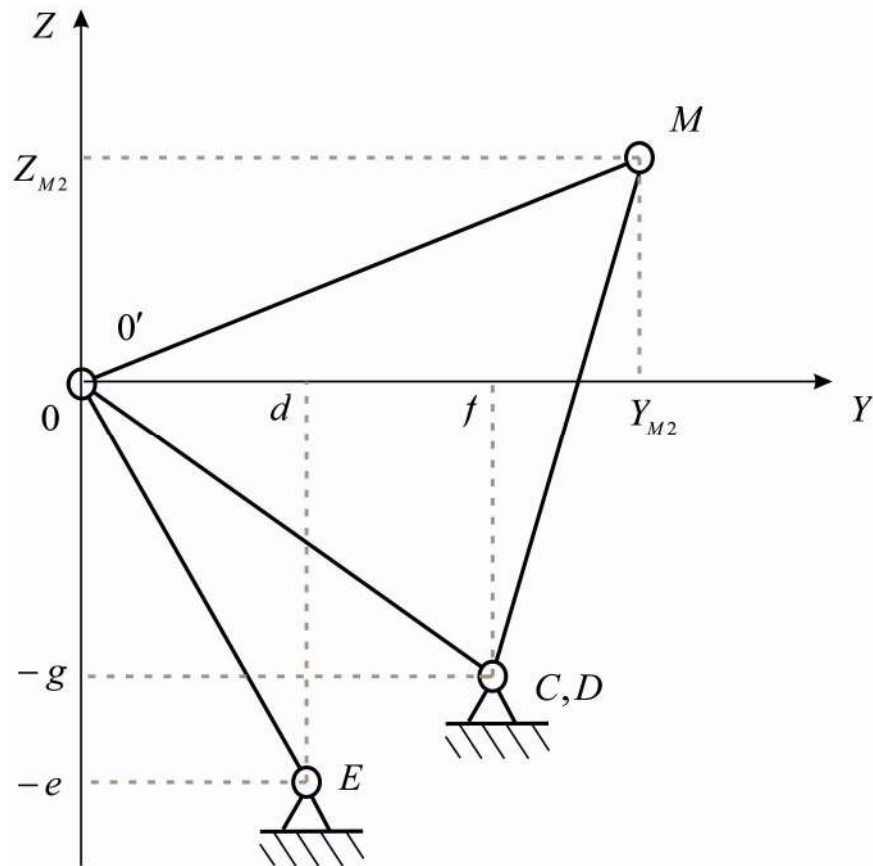


Рисунок 1.21 – Приклад плаского обертання точки M навколо точки C

Математичне уявлення плаского обертання точки навколо точки M навколо точки C , за рахунок зміни довжини циліндра L_4 (II-ий рух), представимо у (1.32):

$$\begin{cases} X_{M_2} = 0; \\ (Y_{M_2} - f)^2 + (Z_{M_2} + g)^2 = MC^2; \\ (Y'_0 - d)^2 + (Z'_0 + e)^2 = L_4; \\ (Y'_0 - f)^2 + (Z'_0 + g)^2 = O'C^2; \\ (Y_{M_2} - Y'_0)^2 + (Z_{M_2} - Z'_0)^2 = O'M^2. \end{cases} \quad (1.32)$$

де d, e, f, g – координати точок кріплення циліндра та поворотної основи;

$O'C = const$ – відстань, що визначається конструкцією поворотної основи;

MC і $O'M$ – відстань, що визначаються конструкцією маніпулятора, взяті при $L_i = L_{i\min}$ ($i = 1, 2, 3$), де $L_{\min}$ – мінімальна довжина i -го циліндра.

Рішення (1.32) дає можливість отримати координати точки $M(X_{M2}, Y_{M2}, Z_{M2})$ від L_2 координат точок кріплення четвертого (поворотного) циліндра та поворотної основи, його можна подати у вигляді наступних систем:

$$\begin{cases} X_{M2} = 0; \\ Y_{M2} = f \pm \sqrt{MC^2 - (Z_{M2} + g)^2} \\ Z_{M2} = \frac{F_3 \pm \sqrt{F_3^2 + 4K_2F_4}}{2K_2} \end{cases} \quad (1.33)$$

де

$$\begin{aligned} F_3 &= 4 \left[O'C^2 \left(\frac{F_1 \pm \sqrt{F_1^2 + 4K_1F_2}}{2K_1} - g \right) + (MC^2 - O'M^2) \cdot \left(\frac{F_1 \pm \sqrt{F_1^2 + 4K_1F_2}}{2K_1} + g \right) \right] \\ F_4 &= 4(MC^2 - g^2) \cdot \left[O'M^2 - \left(\frac{F_1 \pm \sqrt{F_1^2 + 4K_1F_2}}{2K_1} \right)^2 \right] - (O'C^2 - O'M^2 - \\ &\quad - MC^2 - g \frac{F_1 \pm \sqrt{F_1^2 + 4K_1F_2}}{2K_1})^2, \\ F_2 &= 4(O'C - g^2)(L_4^2 - e^2) - [(d - f)^2 - (O'C^2 - g^2 + L_4^2 - e^2)]^2, \\ F_1 &= 4[(g + e) - [(g - e)^2 - (d - f)^2](g - e)(O'C^2 - L_4^2)], \\ K_2 &= 4 \cdot O'C^2, \\ K_1 &= 4[(d - f)^2 + (g - e)^2] \end{aligned}$$

Лінійне переміщення точки M , за рахунок руху шасі мобільного робота (III-й рух), представлене на рис. 1.22.

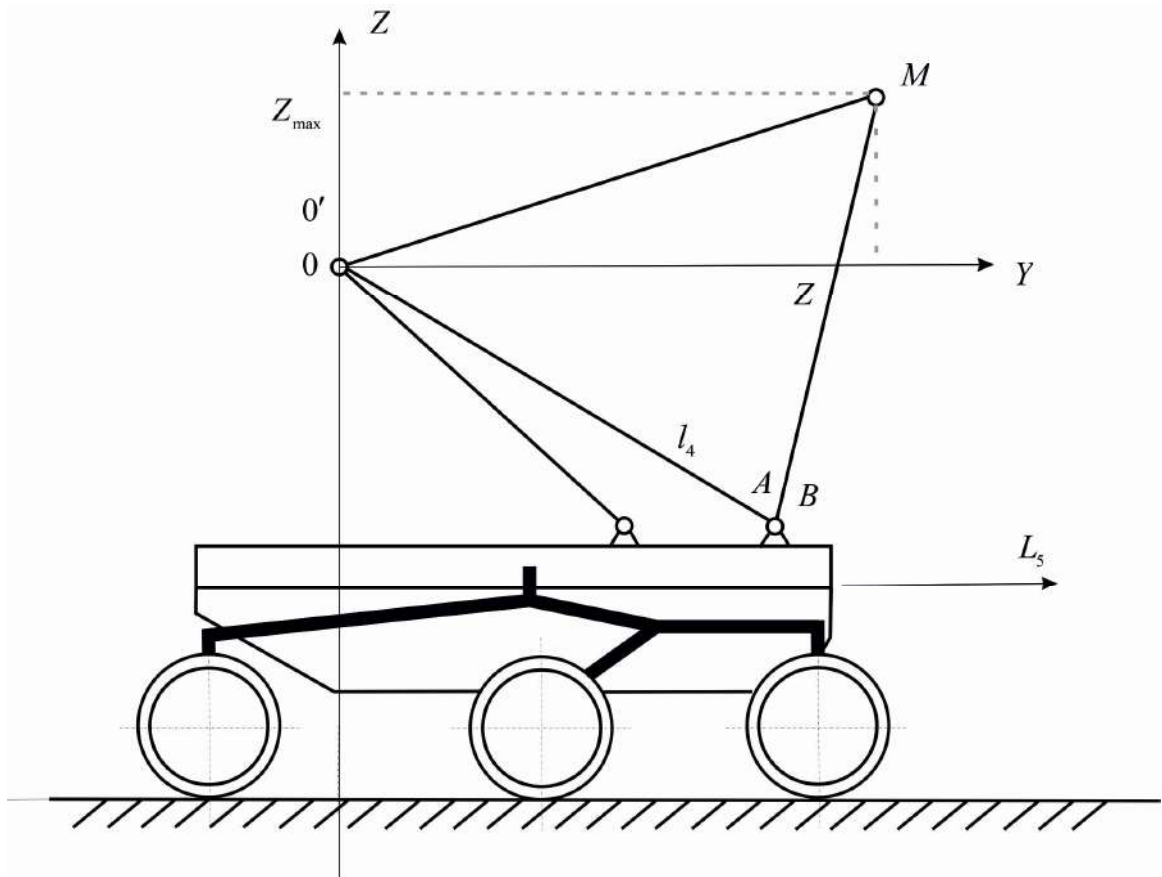


Рисунок 1.22 – Визначення лінійного переміщення точки M

Математичне представлення лінійного переміщення точки M можна представити у вигляді наступної системи:

$$\begin{cases} X_{M3} = 0; \\ Y_{M3} = L_5; \\ Z_{M3} = 0. \end{cases} \quad (1.34)$$

де L_5 – переміщення шасі мобільного робота.

Підставляючи отримані залежності (1.31), (1.32) та (1.34) в рівняння (1.19) отримаємо таку систему:

Отримана система рівнянь повністю визначає теоретичну область кінематично можливих переміщень точки M просторі, тобто теоретичну зону роботи гідроманіпулятора, представлену на рис. 1.9.

$$\begin{cases} X_m = \frac{L_3^2 - L_2^2}{4a}; \\ Y_M = \frac{-abX + c\sqrt{Y}}{4a(b^2 + c^2)} + f \pm \sqrt{MC^2 - \left[\frac{F_3 \pm \sqrt{F_3^2 + 4K_2F_4}}{2K_2} + g \right]^2} + L_5; \\ Z_M = \frac{acX + b\sqrt{Y}}{4a(b^2 + c^2)} + \frac{F_3 \pm \sqrt{F_3^2 + 4K_2F_4}}{2K_2}. \end{cases} \quad (1.35)$$

Зазначена зона є графічним відображенням рішення системи (1.35) і утворюється наступним чином: при нерухомому шасі і крайньому нижньому положенні поворотної основи зона можливих переміщень точки M (вершини піраміди, утвореної трьома циліндрами L_1, L_2, L_3) являє собою об'єм фігури $M_1 \dots M_8$, утвореної перетином шести сферичних поверхонь радіусом $L_{\min}$ і $L_{\max}$, що мають центри послідовно в точках O, A і B , де $L_{\min}$ і $L_{\max}$ – відповідно мінімальна і максимальна довжина кожного циліндра L_1, L_2, L_3 .

Тут M_n ($n=1,2,\dots,8$) характерні положення точки M , по досягненню яких подальше переміщення вантажу можливо здійснити зміною напрямку переміщення штока циліндра, що працює, або включенням в роботу іншого циліндра.

Додаткова робоча зона утворюється при розвороті основи за допомогою циліндра L_4 . І тут фігура $M_1 \dots M_8$ розгортається щодо осі C, D (рис. 1.21), займаючи положення $M_9 \dots M_{16}$, при переміщенні шасі відбувається подальше розширення зони роботи гидроманипулятора.

Зображена на рис 1.23 складна геометрична фігура $M_1 \dots M_{32}$ ($n=1,2,3,\dots,32$) і є кінематично можливою зоною переміщення точки M , що несе на собі схват. У межах цієї фігури можливе переміщення вершини піраміди у будь-якому напрямку у просторі.

Для порівняння кінематичних можливостей маніпулятора з іншими механізмами позиціонування необхідно отримані значення координат точки M виразити через кути повороту верхнього циліндра L_1 , який можна

вважати стрілою, що має змінну довжину. Позначимо кути повороту цієї стріли у вертикальній і горизонтальній площинах відповідно через φ і ψ зобразимо верхній циліндр L_1 у вигляді відрізків $O'M$ у системі координат $OXYZ$, як представлено на малюнку 1.20.

Розглянутий відрізок має змінну довжину і може займати різні положення у просторі залежно від поєднання довжин L_1, L_2, L_3, L_4 та змінної L_5 . Проте останній чинник, тобто переміщення L_5 не впливає на величину кутів φ і ψ тому його можна не розглядати. Тоді траєкторія точки O' визначається одним обертальним рухом у площині YOZ , а точка M здійснює складний рух, що є сумою обертового в тій же площині та просторового руху в будь-якому напрямку в обсязі фігури $M_1 \dots M_{32}$ (рис. 1.23).

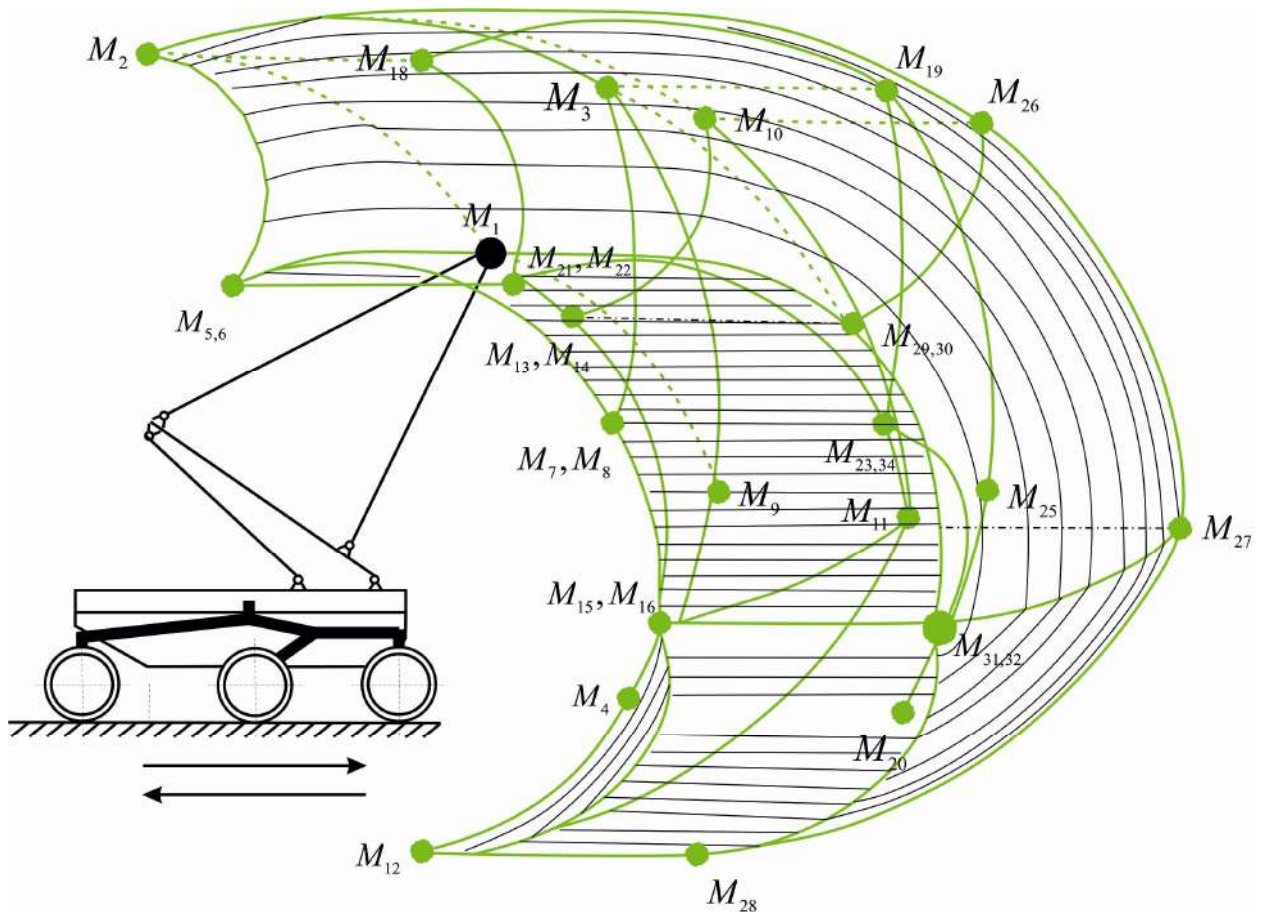


Рисунок 1.23 – Зона дії схвату маніпулятора

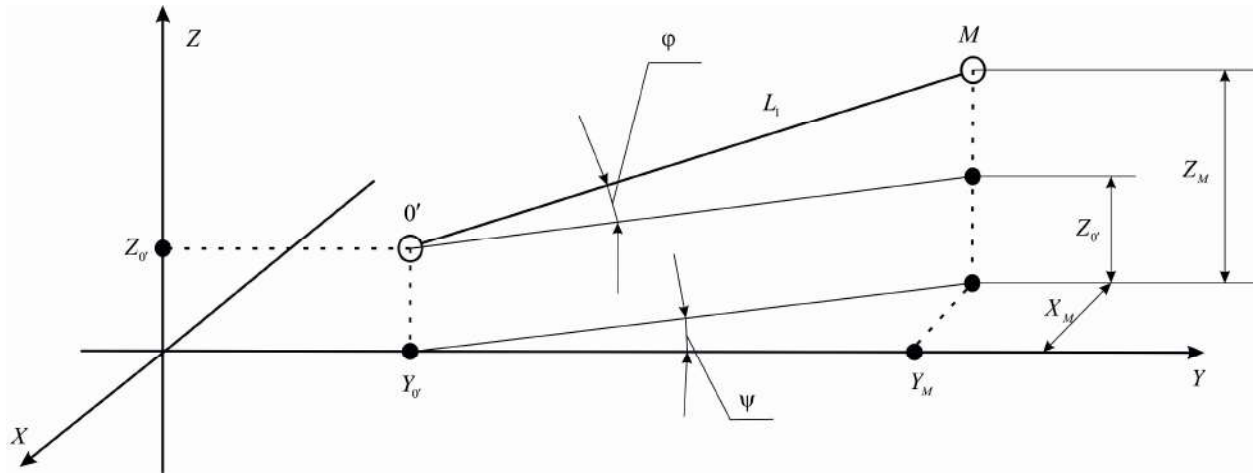


Рисунок 1.24 – До визначення кутів повороту

З рис. 1.24 видно, що шукані кути φ і ψ можна визначити як:

$$\varphi = \arcsin \frac{Z_M - Z_{O'}}{L_1} \quad (1.36)$$

$$\psi = \arctg \frac{X_M X_{O'}}{Y_M - Y_{O'}} \quad (1.37)$$

де $X_M, Y_M, X_{O'}, Y_{O'}, Z_{O'}$ координати точок M і O' визначаються за такими залежностями:

$$\begin{cases} X_m = \frac{L_3^2 - L_2^2}{4a}; \\ Y_M = \frac{-abX + c\sqrt{Y}}{4a(b^2 + c^2)} + f \pm \sqrt{MC^2 - \left[\frac{F_3 \pm \sqrt{F_3^2 + 4K_2 F_4}}{2K_2} + g \right]^2}; \\ Z_M = \frac{acX + b\sqrt{Y}}{4a(b^2 + c^2)} + \frac{F_3 \pm \sqrt{F_3^2 + 4K_2 F_4}}{2K_2}. \end{cases} \quad (1.38)$$

$$\begin{cases} X_{0'} = 0; \\ Y_{0'} = f \pm \sqrt{0'C^2 - \left(\frac{F_1 \pm \sqrt{F_1^2 + 4K_1F_2}}{2K_1} + g\right)^2}; \\ Z_{0'} = \frac{F_1 \pm \sqrt{F_1^2 + 4K_1F_2}}{2K_1} \end{cases} \quad (1.39)$$

Підставляючи значення X_M, Y_M, Z_M і XO', YO', ZO' в (1.36) і (1.37) та провівши відповідні перетворення, отримаємо:

$$\phi = \arcsin \frac{(acX + b\sqrt{Y})K_1K_2 + 2a(b^2 + c^2)\left[(F_3 \pm \sqrt{F_3^2 + 4K_2F_4})K_1\right] - (F_1 \pm \sqrt{F_1^2 + 4K_1F_2})K_2}{L_1} \quad (1.40)$$

$$\begin{aligned} \psi = & \arctg \frac{(L_3^2 - L_2^2)(b^2 + c^2)K_1K_2}{(-abX + c\sqrt{Y})K_1K_2 \pm 2a(b^2 + c^2)} \times \\ & \times \frac{1}{\left[K_1 \sqrt{4Mc^2K_2^2 - (F_3 \pm \sqrt{F_3^2 + 4K_2F_4} + 2gK_2)^2} - \right.} \\ & \left. - K_2 \sqrt{4o'c^2K_1^2 - (F_1 \pm \sqrt{F_1^2 + 4K_1F_2} + 2K_1g)^2} \right] \end{aligned} \quad (1.41)$$

1.3.2 Силевий аналіз маніпулятора

Силевий аналіз усієї конструкції розділений на два етапи: спочатку розглянуто тристрижневу просторову систему, вмонтовану в нерухомій поворотній основі, а потім враховується поворот системи [100–102].

Тоді визначення зусиль у циліндрах зводиться до розкладання зовнішньої сили на три напрями у просторі [103]. Розглянемо загальний випадок, вважаючи що стосовно осей координат зовнішня сила спрямована довільно, як показано рис. 1.25. Цим враховується ухил місцевості, вітрові навантаження та низку інших факторів.

Введемо такі позначення: l_j – поточні значення довжини силових циліндрів; P_j – зусилля на штоках циліндрів; Q_j – проекції зовнішньої сили на осі координат; X_j – координати точки перетину осей симетрії циліндрів у системі координат $OX_1X_2X_3$; a_{ij} – координати точок кріплення корпусів циліндрів виходячи з системи координат $M(x'_1, x'_2, x'_3)$ ($i = j = 1, 2, 3$).

Зовнішня сила додана до вершини M тристрижневого вузла. Сили, що проходять через цю точку, в системі координат $M(x'_1, x'_2, x'_3)$ повинні задовольняти наступній умові:

$$\begin{cases} \lambda_{11}P_1 + \lambda_{12}P_2 + \lambda_{13}P_3 \pm Q_1 = 0; \\ \lambda_{21}P_1 + \lambda_{22}P_2 + \lambda_{23}P_3 \pm Q_1 = 0; \\ \lambda_{31}P_1 + \lambda_{32}P_2 + \lambda_{33}P_3 Q_1 = 0. \end{cases} \quad (1.42)$$

де λ_{ij} – косинуси кутів, що утворюються циліндрами з осями координат напрямних косинусів ($i = j = 1, 2, 3$).

Введемо нові невідомі:

$$N_j = \frac{P_j}{l_i} \quad (1.43)$$

Маючи на увазі відомі співвідношення для визначення напрямних косинусів і скориставшись (1.43) система рівнянь (1.42) перетворюється до виду:

$$\begin{cases} a_{11}N_1 + a_{12}N_2 + a_{13}N_3 \pm Q_1 = 0; \\ a_{21}N_1 + a_{22}N_2 + a_{23}N_3 \pm Q_1 = 0; \\ a_{31}N_1 + a_{32}N_2 + a_{33}N_3 - Q_1 = 0. \end{cases} \quad (1.44)$$

Завдяки цьому вдається уникнути попереднього визначення напрямних косинусів, коефіцієнтами при нових невідомих є безпосередньо координати кінців циліндрів у системі координат $M(x'_1, x'_2, x'_3)$.

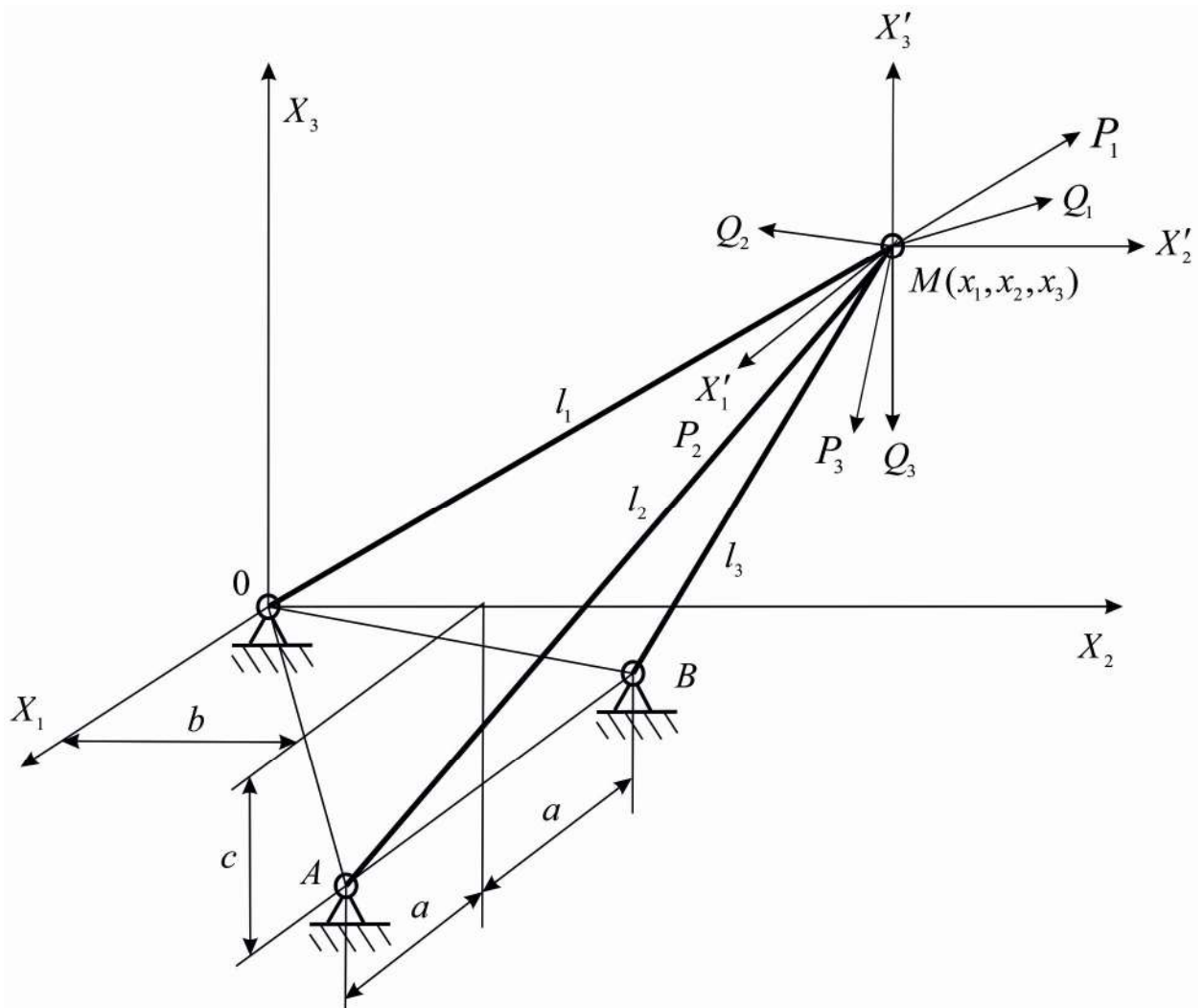


Рисунок 1.25 – До силового аналізу маніпулятора

В результаті розв'язання системи рівнянь (1.44) переходу за допомогою (1.43) та зусиль P_j та повернення до системи координат $OX_1X_2X_3$ вважаючи, що складові Q_1 та Q_2 мають негативний знак, отримаємо:

$$\begin{cases} P_1 = \frac{[(x_2 - b)Q_3 - (x_3 + c)Q_2]l_1}{bx_3 + cx}; \\ P_2 = \frac{[(bx_3 - cx_2)Q_1 + (ax_3 - cx_1)Q_2 - (ax_2 + b)Q_3]l_2}{2a(bx_3 + cx_2)}; \\ P_3 = \frac{[(bx_3 - cx_2)Q_1 + (ax_3 - cx_1)Q_2 + (ax_2 + bx_1)Q_3]l_3}{2a(bx_3 + cx_2)}. \end{cases} \quad (1.45)$$

При інженерних розрахунках вважатимуться, що зовнішня сила спрямовано вертикально: $Q_1 = Q_2 = 0$; $Q_3 = Q$. Для «компенсації» цього припущення, а також для врахування динамічності системи та, зокрема, пов'язаного з цим явища перерозподілу зусиль між циліндрами, введемо відповідні коефіцієнти k_D . Тоді залежності (1.45) набувають вигляду:

$$\begin{cases} P_1 = \frac{x_2 - b}{bx_3 + cx} k_D l_1 Q; \\ P_2 = \frac{ax_2 + bx_1}{2a(bx_3 + cx_2)} k_D l_2 Q; \\ P_3 = \frac{ax_2 - bx_1}{2a(bx_3 + cx_2)} k_D l_3 Q. \end{cases} \quad (1.47)$$

Як відомо з кінематичного аналізу координати вершини $M(x_1, x_2, x_3)$ є функціями довжин силових циліндрів:

$$\begin{cases} x_1 = \frac{l_3^2 - l_2^2}{4a}; \\ x_2 = \frac{-abU + c\sqrt{V}}{4a(b^2 + c^2)}; \\ P_3 = \frac{acU + b\sqrt{V}}{4a(b^2 + c^2)}. \end{cases} \quad (1.48)$$

$$\begin{aligned} \text{де} \quad U &= l_2^2 + l_3^2 - 2l_1 - 2(a^2 + b^2 + c^2); \\ V &= (b^2 + c^2) \cdot [16a^2 l_1^2 - (l_3^2 - l_2^2)^2] - a^2 U^2. \end{aligned}$$

Зусилля P_1, P_2, P_3 також є функціями довжини силових циліндрів і повністю залежать від величини ходу і послідовності переміщень штоків циліндрів, оскільки, у свою чергу $l_j = f(y_j)$, де y_j – поточне значення ходу штока відповідного циліндра. При повороті основи, що несе тристрижневу

просторову систему, відбувається перерозподіл зусиль у циліндрах. Знайдені залежності (1.47) і (1.48) можуть бути справедливими для будь-якого положення основи, якщо під розмірами a, b, c , що визначають місцезнаходження опор A, O, B у рухомій системі координат $OX_1X_2X_3$ (рис. 1.25) мають на увазі їх поточні значення в системі координат $O'X'_1X'_2X'_3$ осі, які є паралельними та однаково орієнтовані по відношенню до базової системи відліку $O_1X_1X_2X_3$. При практичних розрахунках зручніше припускати використовуючи принципи відносності руху, що система є нерухомою, а напрямок дії зовнішньої сили Q змінюється, відхиляючись на кут Θ , рівний куту повороту основи (рис. 1.26).

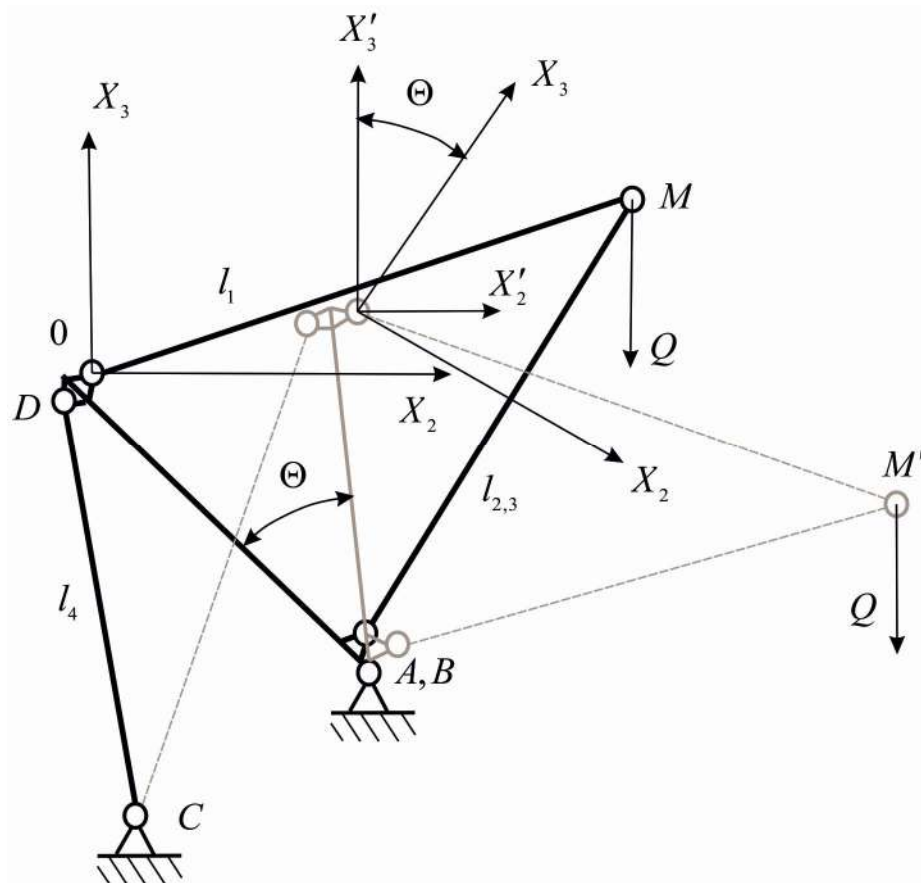


Рисунок 1.16 – Силевий аналіз маніпулятора при повороті його основи

Початок відліку цього кута береться при $l_4 = l_{\min}$ де l_4 – поточне значення довжини циліндра повороту основи. В цьому випадку за основу

приймаються формули (1.47) вважаючи, що при $\Theta=0$ сила Q спрямована вертикально, значення складових по осях координат цієї сили перебувають із співвідношень:

$$Q = 0; \quad Q_2 = Q \sin \Theta; \quad Q_3 = Q \cos \Theta \quad (1.49)$$

У випадку, коли спочатку вища сила задана своїми проекціями Q_j на осі координат, їх значення Q'_j визначаються залежностями:

$$Q'_1 = Q_1; \quad Q'_2 = Q_2 \sin \Theta; \quad Q'_3 = Q_3 \cos \Theta \quad (1.50)$$

В результаті силового аналізу тристрижневої просторової системи, що проводиться за допомогою формул (1.47) і (1.48) встановлено, що зусилля P_1 у верхньому циліндрі 1 (OM) досягає максимального значення при переміщенні вершини M в крайнє нижнє положення, що відповідає наступному поєднанню довжин циліндрів:

$$l_1 = l_{\max}; \quad l_2 = l_{\min}; \quad l_3 = l_{\min} \quad (1.51)$$

При нахилі основи за допомогою циліндра 4(CD) відбувається подальше зростання зусилля P_1 . Максимальні значення зусилля P_2 і P_3 в нижніх циліндрах мають місце при переміщенні вершини M в горизонтальній площині, коли довжини циліндрів визначаються залежностями:

$$l_1 = l_{\min}; \quad l_2 = l_{\min}; \quad l_3 = l_{\max} \quad \text{или} \quad l_1 = l_{\min}; \quad l_2 = l_{\max}; \quad l_3 = l_{\min}. \quad (1.52)$$

При цьому якщо $P_2 = P_{2\max}$, то $P_3 = P_{3\min}$ і навпаки. Після вирішення головного завдання силового аналізу – визначення зусиль у циліндрах, подальший розрахунок системи провадиться відомими способами [104]. Для

знаходження зусиль у циліндрі 4 (CD) Повороту основи складаються звичайні рівняння рівноваги [105].

2 СИСТЕМИ КЕРУВАННЯ МОБІЛЬНИХ РОБОТІВ НА БАЗІ ОДНОПЛАТНИХ КОМП'ЮТЕРІВ ТА МІКРОПРОЦЕСОРІВ

2.1 Характеристики одноплатних комп'ютерів та мікропроцесорів

І мікропроцесор і мікроконтролер призначені для виконання деяких операцій – вони вилучають команди з пам'яті і виконують ці інструкції (що є арифметичними або логічними операціями) і результат використовується для обслуговування вихідних пристроїв. І мікроконтролер і мікропроцесор здатні безперервно проводити вибірку команд із пам'яті та виконувати ці інструкції, поки на пристрій подається живлення. Інструкції являють собою набори бітів. Ці інструкції завжди виймаються з місця їх зберігання, яке називається пам'яттю.

Мікропроцесор (MPU – Micro Processor Unit) містить функціонал комп'ютерного центрального процесора, або ЦП (CPU – Central Processing Unit) на одному напівпровідниковому кристалі (ІМС – інтегральна мікросхема (IC – Integrated Circuit)). За своєю суттю – це мікрокомп'ютер, який використовується для виконання арифметичних та логічних операцій, управління системами, зберігання даних та інших [106–108]. Мікропроцесор обробляє дані, що надходять з вхідних периферійних пристроїв та передає оброблені дані на вихідні периферійні пристрої.

Існує чотири основні типи процесорів, що відрізняються своєю архітектурою:

- мікропроцесори з повним набором команд (Complex Instruction Set Computer, CISC-архітектура) [109, 110] характеризуються нефіксованим значенням довжини команди, кодуванням арифметичних дій однією командою, невеликою кількістю регістрів, що виконують суворо певні функції. Прикладом такого типу процесорів є сімейство x86;

– мікропроцесори зі скороченим набором команд (Reduced Instruction Set Computer, RISC-архітектура) мають, як правило, підвищену швидкодію за рахунок спрощення інструкцій, що дозволяє спростити процес декодування і, відповідно, скоротити час їх виконання [109, 111]. Більшість графічних процесорів розробляються, використовуючи цей тип архітектури;

– мікропроцесори з мінімальним набором команд (Minimal Instruction Set Computer, MISC-архітектура) [112,113], на відміну від RISC-архітектури, використовують довгі командні слова, що дозволяє виконувати досить складні дії за цикл роботи пристрою. Формування довгих «командних слів» стало можливим завдяки збільшенню розрядності мікропроцесорних пристроїв;

– суперскалярні процесори (Superscalar Processors) використовують кілька декодерів команд, які завантажують роботою безліч виконавчих блоків [114]. Планування виконання потоку команд відбувається динамічно та здійснюється самим обчислювальним ядром. Приклад процесора з таким типом архітектури є, наприклад Cortex A8;

Окремо можна назвати мікропроцесори спеціального призначення (ASIC – Application Specific Integrated Circuit). Як випливає з назви, вони призначені для вирішення конкретного завдання. На відміну від мікропроцесорів загального призначення застосовуються в конкретному пристрої і виконують певні функції, характерні тільки для даного пристрою [115,116]. Спеціалізація на виконанні вузького класу функцій призводить до збільшення швидкості роботи пристрою, як правило, дозволяє знизити вартість такої інтегральної схеми. Прикладами таких мікропроцесорів може бути мікросхема, розроблена виключно для управління мобільним телефоном, мікросхеми апаратного кодування та декодування аудіо- та відеосигналів – так звані цифрові сигнальні процесори (Digital Signal Processing, DSP multiprocessors), які можуть бути реалізовані як ПЛІС (програмована логічна інтегральна схема). При розробці таких процесорів, для опису їх функціональності, використовують мови опису апаратних

пристроїв (HDL – Hardware Description Language), такі як Verilog та VHDL [117–119].

Системи на основі мікропроцесорів будують приблизно таким чином, як представлено на рис. 2.1.

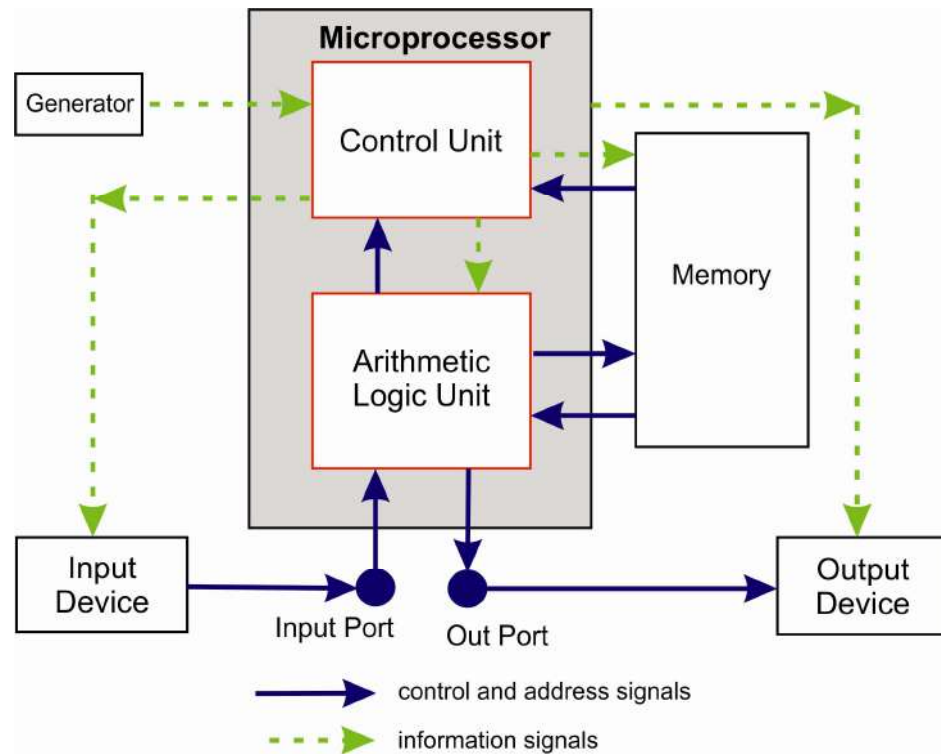


Рисунок 2.1 – Укрупнена архітектура системи на основі мікропроцесора

Як видно, мікропроцесор у цій системі має безліч допоміжних пристроїв, таких як постійний пристрій, оперативна пам'ять, послідовний інтерфейс, таймер, порти вводу/виводу і т.д. Всі ці пристрої обмінюються командами та даними з мікропроцесором через системну шину. Всі допоміжні пристрої у мікропроцесорній системі є зовнішніми. Системна шина, у свою чергу, складається з адресної шини, шини даних та шини керування.

Мікроконтролер відрізняється від мікропроцесора тим, що всі опорні пристрої, такі як постійний запам'ятовуючий пристрій, оперативна пам'ять, таймер, послідовний інтерфейс, порти введення/виводу є вбудованими [120–

122]. Тому не виникає потреби створювати інтерфейси з цими допоміжними пристроями, і це заощаджує багато часу для розробника системи. Приклад структури мікроконтролера наведено рис. 2.2

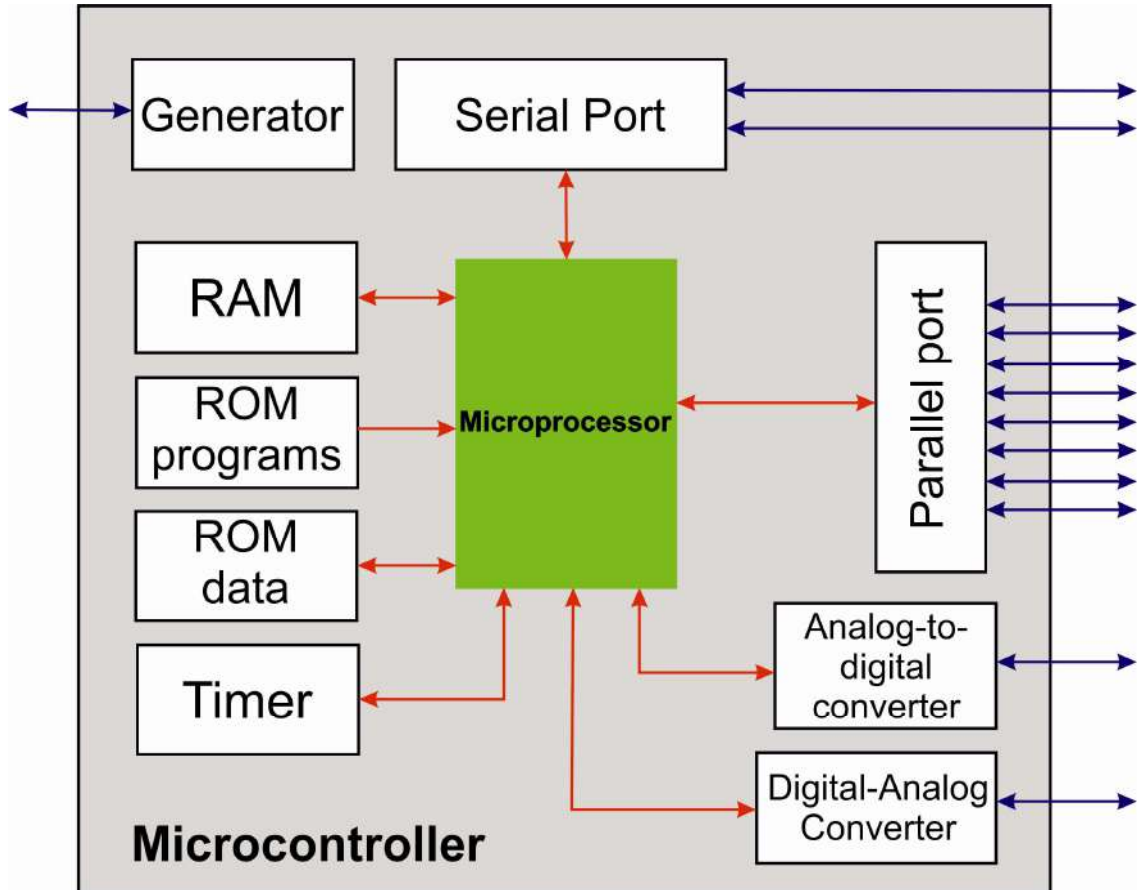


Рисунок 2.2 – Укрупнена структурна мікроконтролера

Мікроконтролер не що інше, як мікропроцесорна система з усіма опорними пристроями, інтегрованими в одному чипі. Якщо необхідно створити пристрій, що взаємодіє із зовнішньою пам'яттю або блоком ЦАП/АЦП, потрібно лише підключити відповідне джерело живлення постійної напруги, ланцюг скидання та кристал кварцу (джерело тактової частоти), які проблематично інтегрувати у напівпровідниковий кристал.

Програма, записана в пам'ять мікроконтролера, може бути захищена від можливості її подальшого читання/запису, що забезпечує захист від її несанкціонованого використання.

У таблиці 2.1 наведено аналіз порівняння мікроконтролера та мікропроцесора [123–125].

Таблиця 2.1 – Порівняння параметрів мікроконтролера та мікропроцесора

Характеристика	Мікропроцесор	Мікроконтролер
Використання	Комп'ютерні системи	Вбудовувані системи
Пристрій	Містить центральний процесор, реєстри загального призначення, покажчики стека, лічильники програми, таймер та ланцюги переривань	Містить схему мікропроцесора і має вбудовані ПЗП, ОЗП, пристрої вводу/виводу, таймери та лічильники
Пам'ять даних	Має багато інструкцій для переміщення даних між пам'яттю та процесором	Має одну- дві інструкції для переміщення даних між пам'яттю та процесором
Електричні ланцюги	Висока складність	Досить прості
Витрати	Вартість всієї системи збільшується	Низька вартість системи
Число реєстрів	Має меншу кількість реєстрів, операції переважно виробляються в пам'яті	Має більше реєстрів, тому простіше писати програми
Запам'ятовуючий пристрій	Заснований на архітектурі фон Неймана. Програма та дані зберігаються у тому ж модулі пам'яті	Засновано на Гарвардській архітектурі. Програми та дані зберігаються у різних модулях пам'яті
Час доступу	Час доступу до пам'яті та пристроїв введення/виводу більше	Найменший час доступу для вбудованої пам'яті та пристроїв для введення/виводу
Залізо	Вимагає більше апаратного забезпечення	Вимагає менше апаратного забезпечення

SRAM – статична пам'ять з довільним доступом (SRAM, static random access memory) – напівпровідникова оперативна пам'ять, в якій кожен двійковий або трійковий розряд зберігається у схемі з позитивним зворотним зв'язком, що дозволяє підтримувати стан без регенерації, необхідної в динамічній пам'яті (DRAM) [126,12]. Проте зберігати дані без перезапису SRAM можна тільки поки є живлення, тобто SRAM залишається енергозалежним типом пам'яті.

EEPROM – (англ. Electrically Erasable Programmable Read-Only Memory) – електрично стираємий ПЗП (ЕСПЗП), що перепрограмується, один з видів енергонезалежної пам'яті (таких, як PROM і EPROM). Пам'ять такого типу може стиратися та заповнюватися даними до мільйона разів.

DIGITAL – цифровий сигнал, який можна подати у вигляді послідовності дискретних (цифрових) значень. У наш час найпоширеніші двійкові цифрові сигнали (бітовий потік) у зв'язку з простотою кодування та використанням у двійковій електроніці [128, 129].

Використання в цифрових системах алгоритмів перевірки та відновлення цифрової інформації дозволяє суттєво збільшити надійність передачі.

ANALOG – аналоговий сигнал (сигнал даних), у якого кожен з параметрів описується функцією часу і безперервною безліччю можливих значень.

Властивості аналогових сигналів значною мірою є протилежністю властивостей квантованих чи цифрових сигналів [130, 131].

Відсутність чіткої візничі один від одного дискретних рівнів сигналу призводить до неможливості застосувати для його опису поняття інформації у вигляді, як вона розуміється в цифрових технологіях. «Кількість інформації», що міститься в одному відліку, буде обмеженою лише динамічним діапазоном засобу вимірювання.

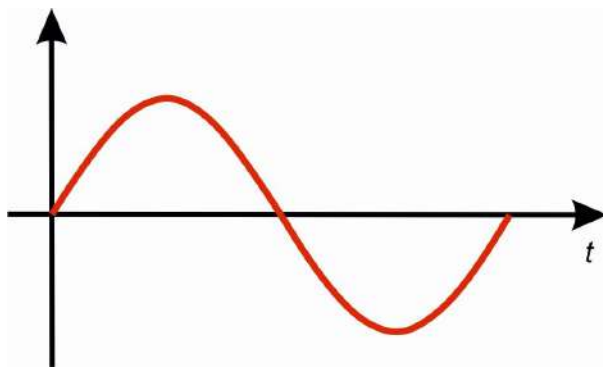
Відсутність надмірності. З безперервності простору значень випливає, що будь-яка завада, внесена в сигнал, не відрізняється від самого сигналу і,

отже, вихідна амплітуда може бути відновлена. Насправді, фільтрація можлива, наприклад, частотними методами, якщо відома будь-яка додаткова інформація про властивості цього сигналу (зокрема, смуга частот).

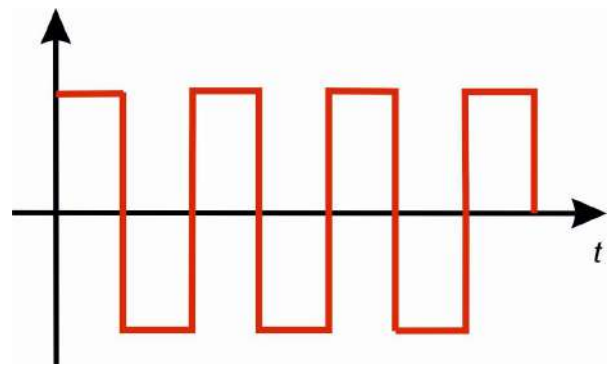
Аналогові сигнали часто використовують для представлення фізичних величин, що безперервно змінюються. Наприклад, аналоговий електричний сигнал, що знімається з термопари, несе інформацію про зміну температури, сигнал з мікрофона — про швидкі зміни тиску в звуковій хвилі тощо.

DISCRETE – дискретний сигнал, який є переривчастим, на відміну від аналогового, і який змінюється в часі та набуває будь-якого значення зі списку можливих значень [132, 133]. При цьому він змінюється або лише в часі, залишаючись безперервним за величиною, або лише за рівнем, будучи безперервним за часом. Що є відмінністю від цифрового сигналу, під яким розуміють дискретний квантований сигнал за рівнем.

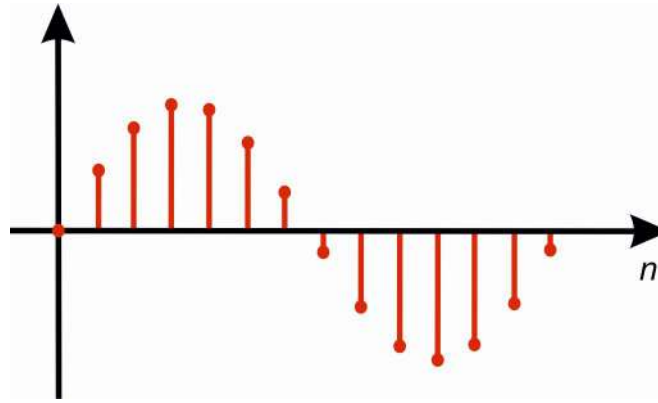
Дискретність застосовується в обчислювальній техніці для пакетної передачі даних. На рис. 2.3 представлено приклад аналогового, цифрового та дискретного сигналу.



а) аналоговий сигнал



б) цифровий сигнал



в) дискретний сигнал

Рисунок 2.3 – Форми сигналів

5V – на вивід надходить напруга 5В від стабілізатора напруги на платі, незалежно від того, як живиться пристрій: від адаптера (7 – 12В), від USB (5В) або через вивід VIN (7 – 12В). Живлення пристрою через виводи 5V або 3V3 не рекомендується, оскільки в цьому випадку не використовується стабілізатор напруги, що може призвести до виходу його з ладу.

3V3 – 3.3В, що надходять від стабілізатора напруги на платі. Максимальний струм, який споживається від цього виводу, становить 50 мА.

GND – виводи землі.

SERIAL – (англ. Universal Asynchronous Receiver-Transmitter) всі плати Arduino мають принаймні один послідовний порт (також відомий як UART або USART): Serial [134–136]. Він пов'язаний із цифровими виводами 0 (RX) та 1 (TX), а також використовується для зв'язку з комп'ютером через USB. Таким чином, під час використання послідовного порту, виводи 0 і 1 не можуть використовуватися як цифрові входи або виходи. Для зв'язку з Arduino можна використовувати спеціальну програму моніторингу послідовного порту, вбудовану в програмне забезпечення Ардуїно.

SPI (англ. Serial Peripheral Interface), SPI bus – послідовний периферійний інтерфейс; шина SPI – послідовний синхронний стандарт передачі даних у режимі повного дуплексу, призначений для забезпечення

простого та недорогого високошвидкісного сполучення мікроконтролерів та периферії. SPI також іноді називають чотирипровідним (англ. four-wire) інтерфейсом [137–139].

На відміну від стандартного послідовного порту (англ. standard serial port), SPI є синхронним інтерфейсом, у якому будь-яка передача синхронізована із загальним тактовим сигналом, що генерується провідним пристроєм (процесором). Приймаюча (ведена) периферія синхронізує отримання бітової послідовності з тактовим сигналом. До одного послідовного інтерфейсу периферійного провідного пристрою-мікросхеми може приєднуватися кілька мікросхем. Провідний пристрій обирає ведене для передачі, активуючи сигнал «вибір кристала» (chip select) на веденій мікросхемі. Периферія, не обрана процесором, не бере участі у передачі SPI.

Зв'язок за інтерфейсом SPI arduino відбувається між декількома пристроями, які розташовані близько один до одного. Плати Ардуїно мають окремі висновки для SPI. Сполучення відбувається за допомогою чотирьох контактів:

- MOSI – цією лінією передається інформація до Slave від Master;
- MISO – використовується передача інформації до Master від Slave;
- SCLK – створення тактових імпульсів для синхронної передачі;
- SS – вибір керованого пристрою.

На рисунку 2.4 наведено метод з'єднання інтерфейсу SPI.

I²C – послідовний протокол обміну даними ІІС (Inter-Integrated Circuits, міжмікросхемне з'єднання) використовує для передачі даних дві двонаправлені лінії зв'язку, які називаються шина послідовних даних SDA (Serial Data) та шина тактування SCL (Serial Clock) [140–142]. Також є дві лінії для живлення. Шини SDA та SCL підтягуються до шини живлення через резистори. У мережі є хоча б один провідний пристрій, який ініціалізує передачу даних і генерує сигнали синхронізації. У мережі також є ведені пристрої (Slave), які передають дані на запит ведучого. Кожен ведений

пристрій має унікальну адресу, за якою ведучий і звертається до нього. Адреса пристрою вказується у паспорті (datasheet).

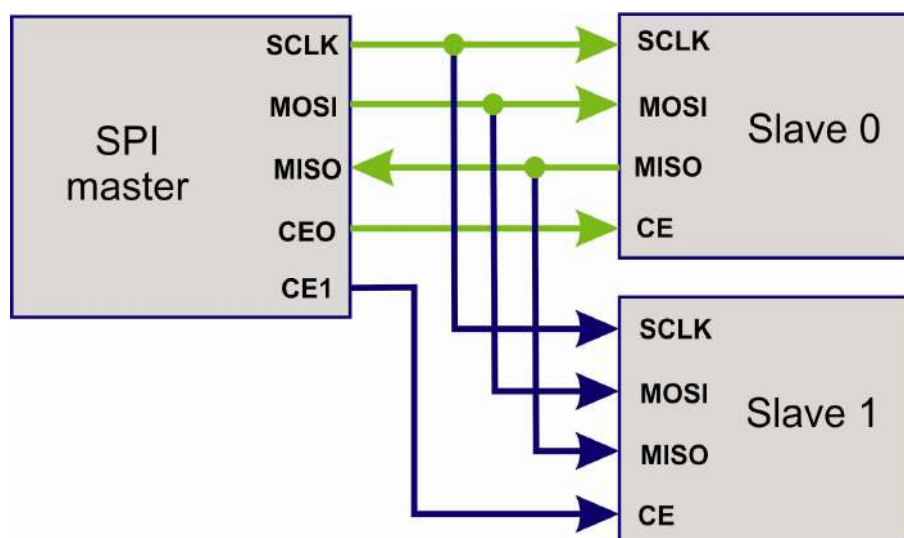


Рисунок 2.4 – Метод з'єднання інтерфейсу SPI

До однієї шини I²C може бути підключено до 127 пристроїв, зокрема кілька провідних. До шини можна підключати пристрої у процесі роботи, тобто вона підтримує гаряче підключення. Приклад підключення наведено рисунку 2.5.

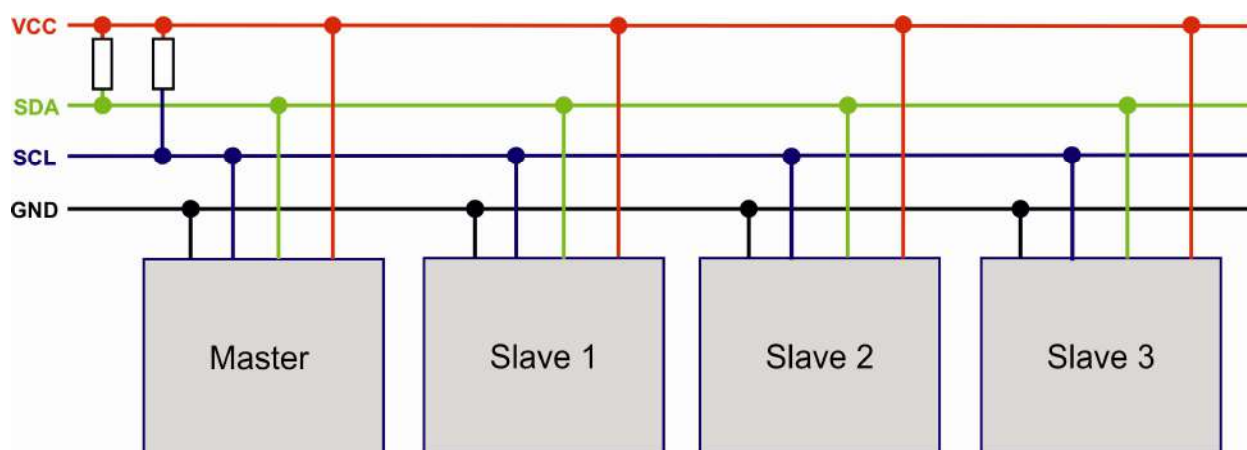


Рисунок 2.5 – Метод з'єднання I²C [142]

PWM, або ШІМ (широтно-імпульсна модуляція) – це операція отримання аналогового значення, що змінюється, за допомогою цифрових

пристроїв. Пристрої використовуються для отримання прямокутних імпульсів – сигналу, який постійно перемикається між максимальним та мінімальним значеннями [143–145]. Цей сигнал моделює напругу між максимальним значенням (5 В) і мінімальним (0 В), змінюючи при цьому тривалість часу включення 5 В щодо включення 0 В. Тривалість включення максимального значення називається шириною імпульсу. Для отримання різноманітних аналогових величин змінюється ширина імпульсу. При досить швидкій зміні періодів включення-вимкнення можна подавати постійний сигнал між 0 і 5 на світлодіод, тим самим керуючи яскравістю його світіння.

На графіку (рис. 2.6) зелені лінії відзначають постійні часові періоди. Тривалість періоду обернено пропорційна частоті ШІМ. Тобто, якщо частота ШІМ становить 500 Гц, то зелені лінії відзначатимуть інтервали тривалістю 2 мілісекунди кожен. Виклик функції `analogWrite()` з масштабом 0 – 255 означає, що значення `analogWrite(255)` буде відповідати 100% робочому циклу (постійне включення 5 В), а значення `analogWrite(127)` – 50% робочому циклу.

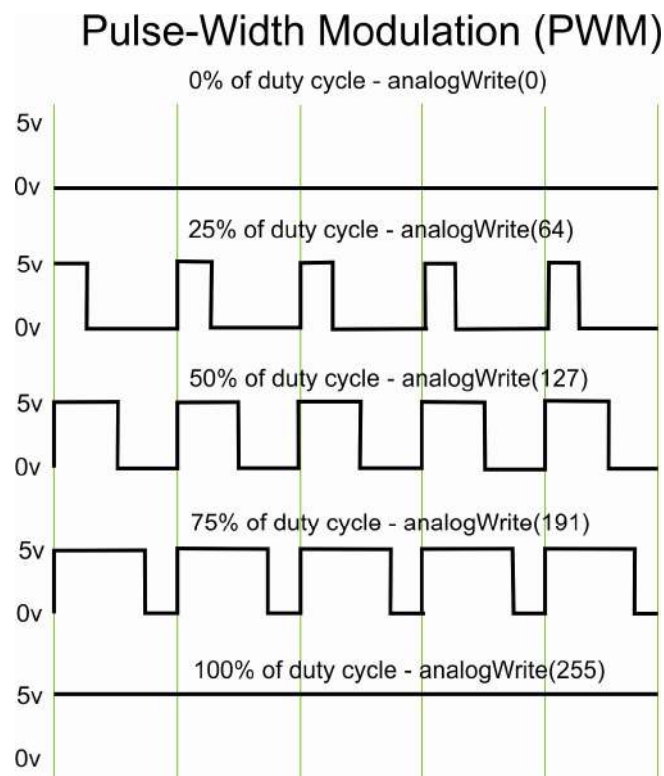


Рисунок 2.6 – ШІМ модуляція

2.2 Asus Tinker Board

Asus Tinker Board 2Гб – являє собою одноплатний комп'ютер в ультра-малому форм-факторі, який пропонує досить високу продуктивність в своєму класі. Плата має на борту потужний ARM Cortex-A17 з тактовою частотою 1,8 ГГц, гігабітний Ethernet, порт HDMI 2.0 для моніторів і телевізорів з роздільною здатністю до 4K і H.264, 2 Гб оперативної пам'яті, звук 192K / 24 біта, SDIO 3.0 і ще кілька бонусів [146–148]. Крім того до плати можна підключити різні модулі розширення, які є у продажу. Загальний вигляд Asus Tinker Board 2Гб представлений на рисунку 2.7, а основні характеристики в таблиці 2.2

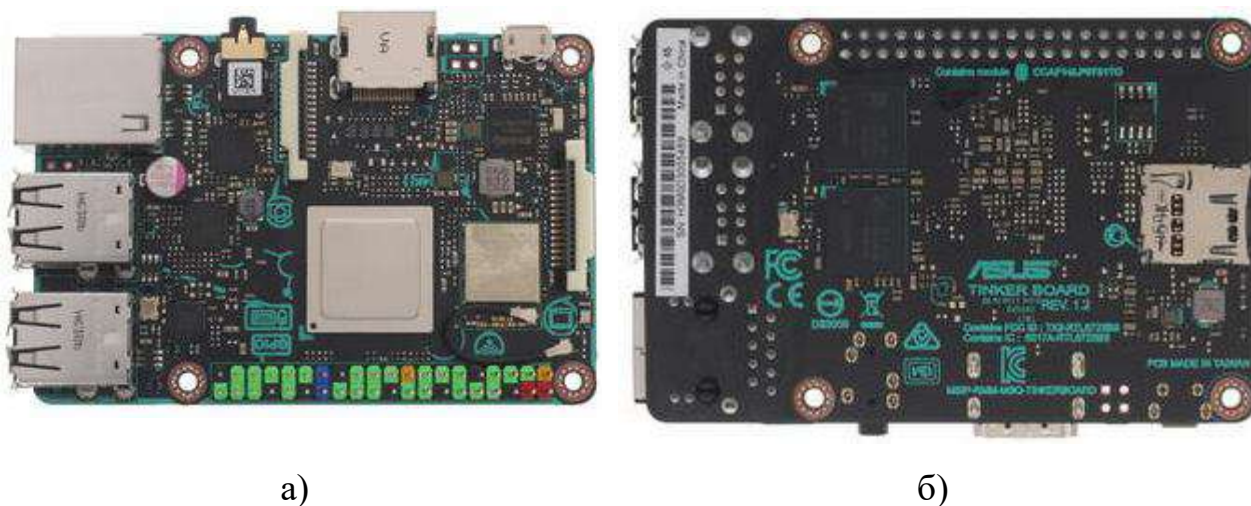


Рисунок 2.7 – Мікрокомп'ютер ASUS Tinker Board[146]

Таблиця 2.2 – Основні характеристики ASUS Tinker Board[147–148]

Параметри	Значення
1	2
Процесор	
Виробник процесора	Rockchip
Модель процесора	Rockchip RK3288

Продовження таблиці 2.2

1	2
Кількість ядер процесора	4
Частота процесора	1800 МГц
Оперативна пам'ять	
Тип оперативної пам'яті	LPDDR3
Розмір оперативної пам'яті	2 Гб
Накопичувачі даних	
Об'єм вбудованої флеш-пам'яті	немає
Відеокарта	
Виробник відеочипа	Mali
Модель відеочипа	Mali T764
Обсяг відеопам'яті	виділяється з оперативної
Інтерфейси, роз'єми	
Відео інтерфейси	HDMI
Аудіо інтерфейси	3.5 мм jack (аудіо)
Інтерфейси периферії	display serial interface, camera serial interface, 40 pin GPIO, USB 2.0 x4
Додаткове вбудоване обладнання	Bluetooth 4.0
Особливості	HDMI 2.0 з підтримкою 4K, слот Micro SD з підтримкою UHS-I
Комунікації	
Вид доступу в Інтернет	Wi-Fi, Ethernet (RJ-45)
Швидкість мережевого адаптера	1000 Мбіт/с
Габарити (Д*Ш)	85.6 мм * 54 мм

Переваги:

- у порівнянні з іншими одноплатними комп'ютерами Orang Pi, Bana Pi і т.п. тут є драйвера, працює апаратне прискорення відео. Open GL ES краще ніж нічого. Простий доступ до GPIO за допомогою WiringPi;
- по залізу добре збалансований, тобто швидкодія всіх компонентів, а не окремих елементів, картрідер читає 65м / с, ОЗУ працює в двоканальному режимі, мережа працює набагато швидше ніж у Raspberry Pi, GPU 80Gflops майже як на ноутбуці 2010 року, на Raspberry Pi тільки 10Gflops. CPU витримує заявлену чистоту, і при цьому не поганий звук при наявності розв'язки з колонкою;
- завантаження стандартної TinkerOS кілька секунд (з флешки на 65мб / с відповідно) теж саме компіляція невеликих скетчів на QT 2-3с. YouTube в HD можна дивитися з браузера (без фризів) думаю з охолодженням можна і FULL HD, а так може іноді пригальмовувати. VLC плеєр 4K відео заявлені в характеристиках ASUS Tinker Board відпрацьовує на мінімальних FPS;
- 15-контактний роз'єм DSI и CSI.

Недоліки:

- гріється до 60-70 град. тому throttling (механізм захисту процесора від перегріву) ріже частоту під навантаженням, штатний радіатор краще замінити на модуль активного охолодження;
- живлення потрібне якісне 2-3А, шнур короткий інакше можливе самовільне перезавантаження;
- активну систему охолодження жити через GPIO не бажано;
- драйвера на TinkerOS Debian дуже «чутливі» в наслідок чого оновлювати образ ОС некваліфікованим користувачам (apt get) не рекомендується;
- TinkerOS Android потребує передвстановлений Google Play Store;
- підтримка API GPIO все ще знаходиться в «зародковому» стані.

2.3 Raspberry Pi Foundation

Raspberry Pi 3 Model A+ 512MB – міні-комп'ютер з процесором ARM на якому може працювати операційна система Linux. Raspberry Pi 3 model A + 512Mb є міні-комп'ютер з 512 МБ ОЗУ, HDMI і аудіовиходом з композитним відеосигналом RCA (через гніздо 3,5 мм аудіовиходу), один порт USB і 40-контактним роз'ємом з кроком 2,54мм, який забезпечує доступ до входів і виходів загального призначення (GPIO). Модель A Raspberry Pi менше і дешевше, ніж модель B Raspberry Pi [149–151]. На платі відсутній Ethernet інтерфейс але додані бездротові інтерфейси Wi-Fi і Bluetooth. Для запуску комп'ютера потрібна карта microSD з операційною системою і блок живлення (не входять в комплект). Мінікомп'ютер Raspberry Pi користується великою популярністю. Безліч різноманітних прикладів проектів та інформації доступні в Інтернеті у вільному доступі. Загальний вигляд Raspberry Pi Model A + 512MB представлений на рисунку 2.8.



Рисунок 2.8 – Raspberry Pi Model A+ 512MB [150]

Плата відрізняється малим споживанням і скромними розмірами, що робить її незамінною для створення мобільних пристроїв з автономним живленням.

Основні характеристики Raspberry Pi Model A + 512MB представлені нижче[150,151]:

- процесор BCM2837B0 64-розрядний Cortex-A53 з тактовою частотою 1.4GHz;
- графічний процесор VideoCore IV;
- ОЗУ 512 МБ;
- пам'ять програм: micro-SD карта пам'яті;
- один USB-порт 2.0;
- повнорозмірний вихід HDMI;
- чотирьох-контактний роз'єм 3,5 мм з аудіовиходом і композитним відеовиходом (PAL або NTSC) ;
- вбудований двохдіапазонний Wi-Fi (2.4GHz / 5GHz) стандарту IEEE 802.11 b / g / n / ac (чіп Cypress CYW43455) ;
- вбудований Bluetooth 4.2 LE (чіп Cypress CYW43455) ;
- 40-контактний GPIO-роз'єм з 0,1-дюймовими штирями, які сумісні з роз'ємами 2 x 20;
- інтерфейс камери (CSI) ;
- інтерфейс дисплею (DSI) ;
- слот для карт Micro SD;
- напруга живлення: 5В;
- напруга логічних рівнів: 3,3В;
- споживаний струм: до 2,5А.

Raspberry Pi 3 Model B+ – від Raspberry Pi Foundation. Нова плата помітно краща за попередню, зібрана на більш потужному 4х-ядерному 64-бітному SoC Broadcom BCM2837B0 процесорі, що працює на частоті 1.4ГГц. Так само компанія оновила бездротові адаптери: Wi-Fi став двох-діапазонним стандартом IEEE 802.11ac, а Bluetooth-4.2 BLE [152, 153]. Гігабітний мережевий адаптер Gigabit Ethernet забезпечує передачу даних до 300 Мб в секунду. Плата виглядає більш досконало і вже не нагадує інженерний зразок. Оперативної пам'яті так і залишилося 1Гб, але

збільшилася частота процесора, швидкість мережевих інтерфейсів і додалася функція мережевого завантаження операційної системи. Загальний вигляд Raspberry Pi 3 Model B+ представлений на рисунку 2.9.

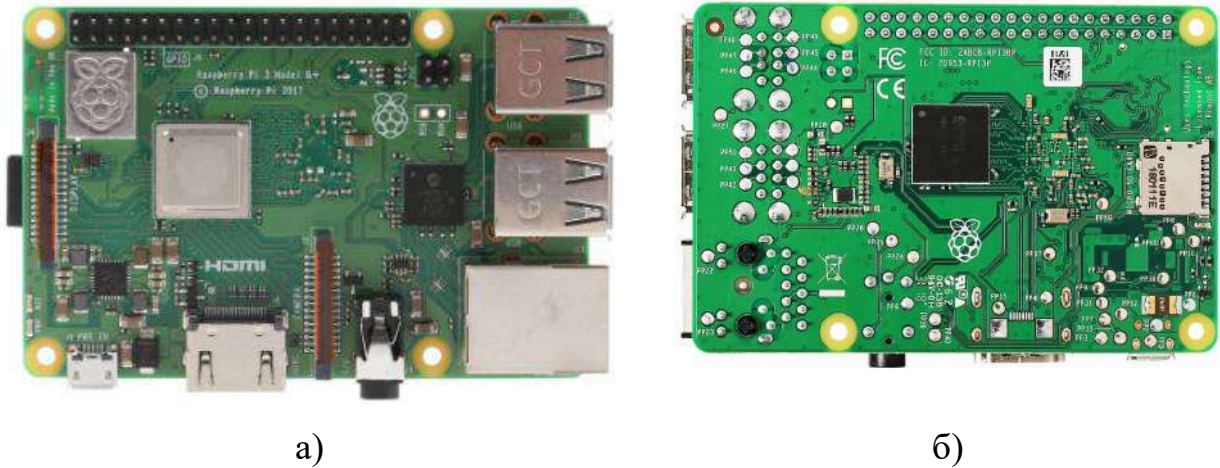


Рисунок 2.9 – Загальний вид Raspberry Pi 3 Model B+ [152]

Для живлення Raspberry обов'язково потрібно застосовувати тільки якісний блок живлення 5 В 3 А. Використання неякісних блоків живлення або зарядних пристроїв мобільних телефонів призводить до виходу з ладу внутрішнього джерела живлення. Максимальне значення вхідної напруги живлення 5,5 В, при вхідній напрузі 6 В (навіть імпульсному) мікросхема перетворювача напруги гарантовано вийде з ладу. До подібного результату призведе замикання джерела напруги 3,3 В на землю.

Основні відмінності від попередньої моделі [153]:

- металева кришка з логотипом Raspberry Pi, що закриває модуль бездротової передачі даних;
- металевий радіатор на процесорі для поліпшеної тепловіддачі;
- 4-піновий роз'єм для підключення PoE.

Процесор Raspberry Pi3B+ при виконанні навіть не дуже важких обчислювальних задач сильно гріється. Забороняється використовувати міні-комп'ютер без додаткового активного охолодження.

Характеристики:

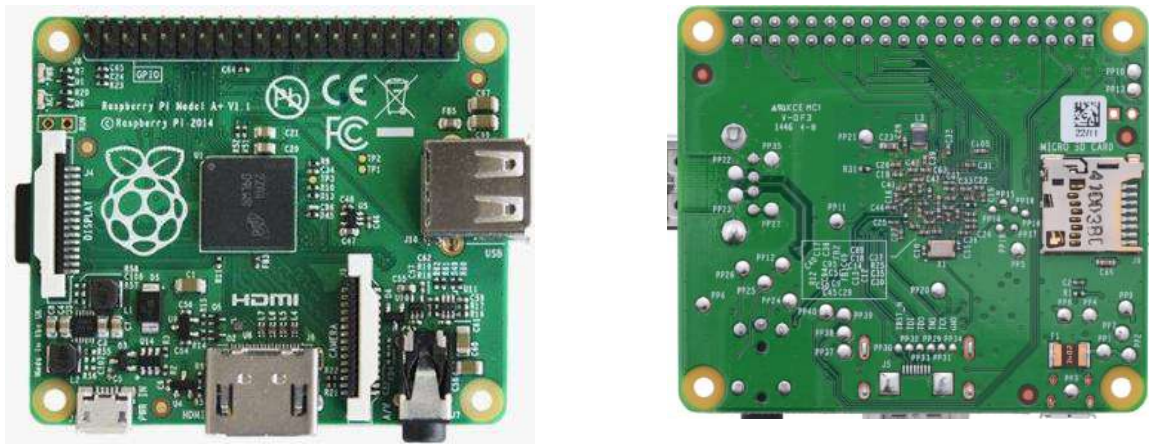
- процесор: SoC Broadcom серії BCM2837B0 (чотириядерний ARM Cortex-A53 1.4GHz з 64-бітною архітектурою);
- оперативна пам'ять: 1 Гб;
- порти USB: 4 USB 2.0 порта;
- Ethernet: гігабітний;
- підтримка живлення по мережі: PoE;
- GPIO: Стандартний 40-піновий роз'єм (з них 26 GPIO);
- роз'єм живлення: MicroUSB;
- двохдіапазонний Wi-Fi (2.4GHz и 5GHz) 802.11ac;
- Bluetooth 4.1 (Classic и Low Energy);
- інтерфейс монітору: HDMI;
- інтерфейс для CSI камери;
- DSI-конектор для LCD дисплею;
- 3.5 мм роз'єм для виведення звуку і композитного відео;
- 2-pin роз'єм для апаратного скидання;
- Micro SD слот.

Основні відмінності від попередньої моделі:

- збільшена тактова частота процесора (до 1,4 ГГц);
- підтримка PoE через окремий роз'єм;
- гігабітний Ethernet
- покращений двохдіапазонний WiFi модуль;
- підтримка PXE (Preboot eXecution Environment) – завантаження ОС через мережевий інтерфейс використання локальних носіїв даних;
- поліпшене керування температурою і потужністю (на базі MaxLinear MxL7704) ;
- поліпшене пасивне охолодження процесора;
- бездротовий модуль, виконаний відповідно до стандарту FCC і захищений металевією кришкою.

Raspberry Pi A+256Mb – це спрощена версія моделі Raspberry Pi Model B+, у якій відсутній порт Ethernet, є тільки один порт USB і 256

мегабайтів оперативної пам'яті. Ці спрощення дозволили ще більше здешевити плату [154, 155]. Маленьке енергоспоживання робить її ідеальною для застосування в проектах, що вимагають від батарей, акумуляторів або сонячних елементів – роботи, сенсори, вилучені застосування або, наприклад, в якості WiFi-репитера. Загальний вигляд Raspberry Pi A+256Mb представлений на рисунку 2.10.



а)

б)

Рисунок 2.10 – Загальний вигляд Raspberry Pi A+256Mb [154]

Основні характеристики представлені нижче:

- процесор 700MHz Broadcom BCM2835 з ОЗУ 256 Мб;
- 40-піновий роз'єм GPIO;
- 1 порт USB 2.0;
- комбінований 4-контактний роз'єм для стерео-аудіовиходу і композитного відео;
- HDMI повного формату;
- порт камери CSI для підключення камери Raspberry Pi;
- цифровий порт дисплея DSI для підключення сенсорного дисплея Raspberry Pi;
- роз'єм Micro SD для завантаження ОС і зберігання даних;
- роз'єм живлення Micro USB;

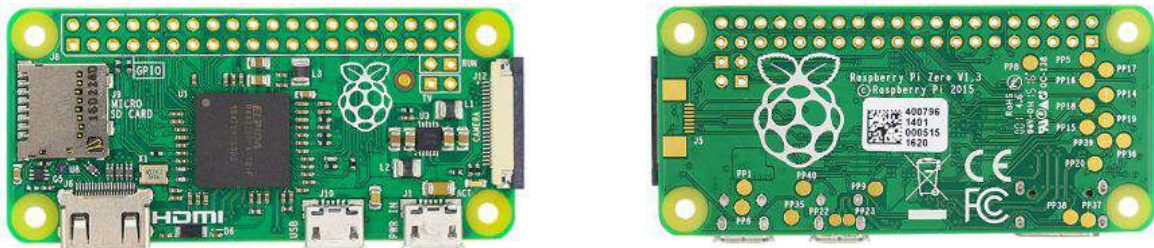
Основні відмінності від Model A:

- ефективніше енергоспоживання;
- поліпшена розводка плати з підтримкою розширеного 40-голкового GPIO;
- порт Micro SD захищає вашу карту від пошкоджень;
- можливість підключення багатоспоживаючих USB пристроїв.

Покращення в Raspberry PI Model A+:

- менший розмір плати і повна сумісність з розширеннями НАТ 40-контактний GPIO;
- можливість підключення камери і сенсорного дисплея;
- потокове відео високої роздільної здатності (1080p) [155].

Raspberry Pi Zero V1.3 – ідеально підходить для створення вбудованих інтернет-проектів Things (IoT). На платі Pi Zero V1.3 компактно розміщені основні міні-роз'єми і нерозпаяні контакти GPIO, що дозволяють використовувати тільки те, що потрібні. В оновленій версії (1.3) додали роз'єм CSI для підключення Pi-відеокамер.[156,157] Загальний вигляд представлений на рисунку 2.11



а)

б)

Рисунок 2.11 – Загальний вид Raspberry Pi Zero V1.3 [156]

Основні характеристики:

- CPU: 1 ГГц процесор ARM11 на чіпі Broadcom BCM2835;
- ОЗУ: 512 МБ DDR2 ELPIDA;
- зовнішня пам'ять: слот карт пам'яті micro-SD;
- міні-HDMI: з вихідним дозволом до 1080@p60;
- Usb-порт: micro-usb з підтримкою On-The-Go (OTG) ;

- живлення: роз'єм USB micro-b для джерела на 5В/2А;
- GPIO: 40-контактний роз'єм;
- додаткові роз'єми: композитне відео, камера CSI.

Raspberry Pi Zero W V1.1 – був розроблений як можна більш гнучким і компактним із гніздами-з'єднувачами-міні-роз'ємами й нерозпаяним 40-контактним GPIO, що дозволяють використовувати тільки те, що вимагає проект [158, 159]. Загальний вид представлено на рисунку 2.12

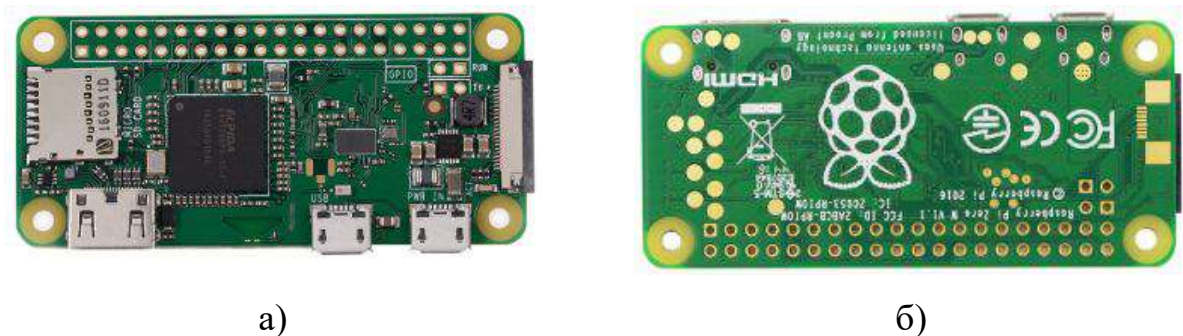


Рисунок 2.12 – Загальний вид Pi Zero W V1.1 [158]

Основні характеристики: [60]

- CPU: 1 ГГц процесор ARM11 на чипі Broadcom BCM2835;
- ОЗУ: 512 МБ DDR2 ELPIDA;
- зовнішня пам'ять: слот карт пам'яті micro-SD;
- міні-HDMI: з вихідним дозволом до 1080@p60;
- USB-порт: micro-USB з підтримкою On-The-Go (OTG) ;
- живлення: роз'єми USB micro-b для джерела на 5В/2А;
- GPIO: 40-контактний роз'єм;
- додаткові роз'єми: композитне відео, камера CSI;
- бездротова мережа: Wi-Fi 802.11 b/g/n;
- Bluetooth: BLE 4.1 на чипі Cypress CYW43438.

Робота Raspberry Pi Zero W V1.1 і Raspberry Pi Zero V1.3 виходить небагато складніше через нестандартні роз'єми. Для початку знадобиться кабель Mini HDMI-HDMI або адаптер для підключення дисплея. Також

знадобиться кабель USB OTG для Usb-обладнань, і ще спеціальний кабель для камери CSI.

У додатку А приведенна зведена таблиця основних параметрів одноплатних комп'ютерів сімейства Raspberry, за допомогою якої можна порівняти і вибрати не обходжену модифікацію для розроблювального проекту.

2.4 Orange Pi

Orange Pi Win Plus A64 Quad-core 2GB WIFI – основна особливість якої – готовність до установки операційної системи Windows 10 Iot на додаток до вже звичних Android, Armbian, Ubuntu, Debian і Raspbian. Як і кожне наступне поповнення комп'ютерів плата доповнена могутнішим процесором Soc Allwinner A64 з 4 64-бітними ядрами ARM Cortex-A53 і GPU Mali-400MP2, розширеним до 2Гб обсягом оперативної пам'яті DDR3 і гнучкою системою живлення, що дозволяє підключати батарейний відсік для забезпечення максимальної автономності [160–161].

Головна відмінність міні-комп'ютера Orange Pi Win Plus від Orange Pi Win полягає в збільшеному до 2Гб обсязі оперативної пам'яті. На платі розведене місце для модуля еммс пам'яті, але сам модуль на платі не розпаяний. У якості пам'яті зберігання програм використовується microSD карта пам'яті з максимальним обсягом до 64Гб.

Для вивіду зображення можна використовувати як MIPI DSI роз'єми для підключення ЖК панелі так і вихід HDMI 1.4, а для підключення камери роздільною здатністю до 5 Мп призначений роз'єм MIPI CSI.

У якості мережних інтерфейсів доступні: мережний порт Gigabit Ethernet, контролер Ampak AP6212 з підтримкою Wi-Fi 802.11n і Bluetooth 4.2. Для підключення додаткових обладнань можна використовувати один microusb порт із підтримкою OTG і чотири порти USB 2.0. 40-контактний GPIO роз'єм сумісний з Raspberry Pi і три UART (один з них

використовується для налагодження), ІК-приймач, кнопки живлення й перезавантаження. Окремий джампер відповідає за вибір джерела живлення: порт micro-USB OTG або зовнішній адаптер з вихідною напругою 5 В, 2 А [160]. Orange Pi Win Plus A64 Quad-core 2GB WIFI представлено на рисунку 2.13.

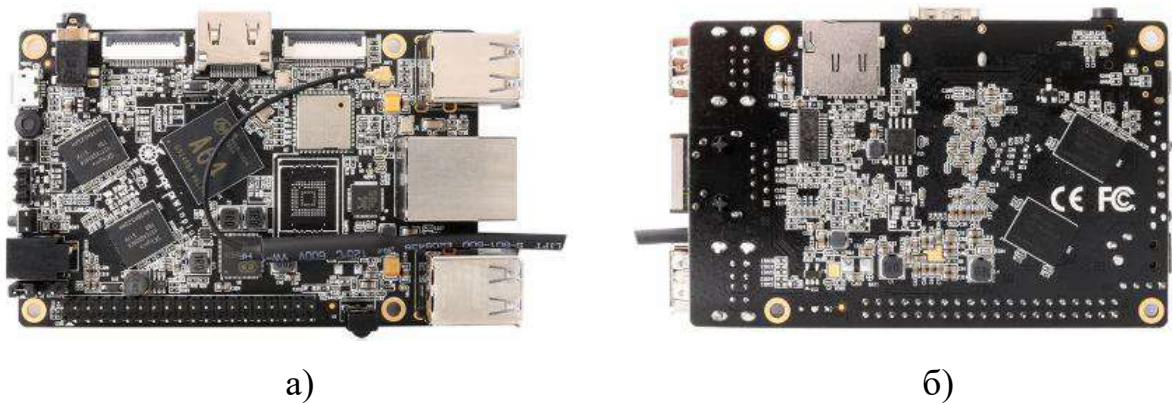


Рисунок 2.13 – Orange Pi Win Plus A64 Quad-core 2GB WIFI [160]

Основні характеристик Orange Pi Win Plus A64 Quad-core 2GB WIFI представлені нижче:

- процесор: Allwinner A64 Quad-core, сімейство процесора: Cortex A53, розрядність процесора: 64-bit, робоча частота: 1,2ГГц–1,6ГГц;
- графічний процесор: Mali400MP2 GPU, підтримка: Opengl ES 2.0 Opengl 1x, DX9_3, Openvg, 1080p45 H.264, апаратна підтримка декодування: H.265/HEVC 4K;
- оперативна пам'ять: 2 ГБ на Win Plus DDR3 (разом з GPU);
- пам'ять програм: microsd (до 64 ГБ), MMC слот, 2M Spi Flash;
- мережні інтерфейси: Ethernet 10/100/1000 Мб/сек Ethernet RJ45, Wi-Fi+Bluetooth Ampak AP6212, IEEE 802.11 b/g/n + Bluetooth 4.2;
- відеовхід: роз'єм 1 x CSI;
- відеовихід: роз'єм HDMI 1.4 з підтримкою HDCP і CEC, роз'єм MIPI LCD;
- аудіовхід: вбудований мікрофон;

- аудіовихід: HDMI, mini-jack (3,5 мм) ;
- USB інтерфейси: USB 2.0 x 4 порту, USB 2.0 OTG x 1 порт;
- інтерфейс дистанційного керування: ІК-порт, інтерфейс ввід-вивід: 40-контактний GPIO сумісний з Raspberry Pi GPIO, UART, I2C, SPI або PWM, UART Інтерфейс (Debug TTL UART) ;
- живлення: напруга живлення: 5В, споживаний струм: 2А;
- тип роз'єму живлення: 4.0 мм/1.7 мм.

Orange Pi Zero 256Mb – це мініатюрний одноплатний комп'ютер на процесорі Allwinner H2+, у який входять чотири обчислювальні ядра Cortex A7 з тактовою частотою до 1,2 ГГц. з відкритим вихідним кодом. Він може працювати під керуванням Android 4.4, Ubuntu, Debian. Він має 256 МБ DDR3 оперативної пам'яті SDRAM. Також цей міні-комп'ютер оснащений графічним прискорювачем Mali-400MP2, який дозволяє декодувати 4 К-відео формату H.265/HEVC із частотою відтворення до 30 кадрів у секунду й підтримує microSD карти пам'яті ємністю до 64 гігабайт [162, 163]. Загальний вид Orange Pi Zero 256Mb представлено на рисунку 2.14.



Рисунок 2.14 – Загальний вид Orange Pi Zero 256Mb [163]

Основні характеристик Orange Pi Zero 256Mb представлені нижче:

- процесор: Soc: Allwinner H2+, Quad-Core Cortex A7@1,2 ГГц;

- графічний процесор: GPU: Mah400MP2(5) 600Mhz, Підтримка Opengl ES2.0;
- оперативна пам'ять: DDR3 RAM: 256M;
- пам'ять програм: microSD;
- мережний інтерфейс: 10/100M Ethernet port;
- USB порти: Host 1x USB 2.0 і 2 порту на додатковім роз'ємі розширення, Micro USB порт: OTG порт, передача даних і вхід живлення;
- відео вихід: композитний;
- інтерфейс камери: відсутній;
- налагоджувальний послідовний порт: 3-х контактний із кроком 2.54ммv;
- GPIO входи/виходи: 2.54мм 24-контактний і 12-контактний роз'єм, включаючи USB, UART, SPI, I2C, PWM і т.д. ;
- напруга живлення: 5 В;
- максимальний споживаний струм: 2 А [163].

Міні-комп'ютер **Orange Pi Zero** відмінно підходить для побудови систем, що вбудовуються, керування й систем контролю. Підтримує наступні операційні системи: Ubuntu MATE, Debian, Armbian і інші. Живиться комп'ютер від блоку живлення 5В 2А через microUSB роз'єм.

Orange Pi PC Plus – це багатофункціональний і потужний одноплатний міні-комп'ютер з відкритим вихідним кодом. Він використовує Allwinner H3 SOC, і має 1 ГБ DDR3 SDRAM. Він може працювати під керуванням Android 4.4, Ubuntu, Debian, Raspberry OS [164]. На платі комп'ютера є 8Гб flash EMMC пам'яті програм і слот для підключення microSD карти пам'яті. Відмінною рисою комп'ютера є те, що при збереженні функціональності й сумісності вийшов комп'ютер дуже малих розмірів. Міні-комп'ютер дуже вимогливий до якості живлення, тому рекомендуємо ретельно підійти до питання живлення й здобувати якісні й надійні блоки живлення [165]. Загальний вид Orange Pi PC Plus представлено на рис. 2.15.

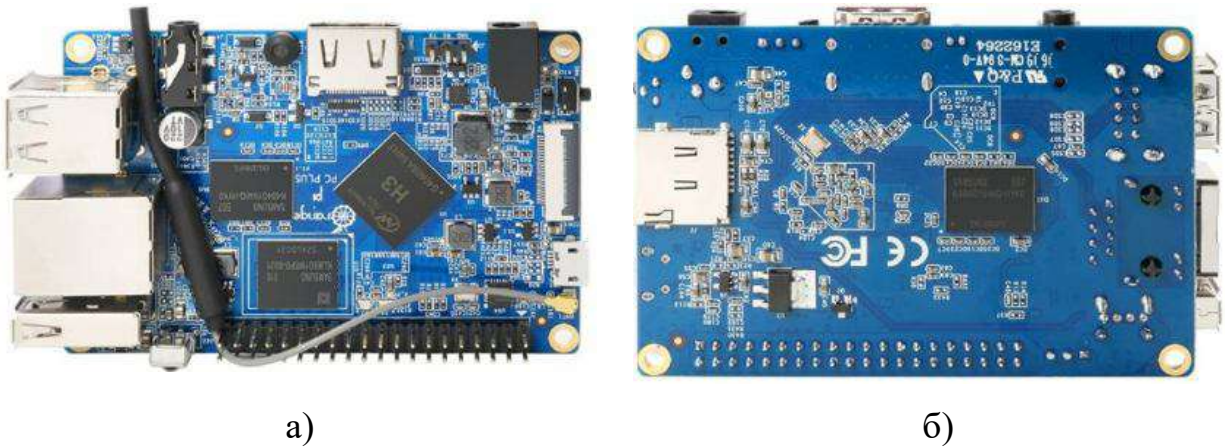


Рисунок 2.15 – Загальний вид Orange Pi PC Plus [165]

Основні характеристики Orange Pi PC Plus:

- процесор: НЗ чотириядерний процесор Cortex-A7 H.265/HEVC 4 до 1.6ГГц;
- графічний процесор: Mali400MP2 GPU @ 600 МГц із підтримкою OpenGL ES 2.0;
- оперативна пам'ять: SDRAM 1 ГБ DDR3 (використовується разом з GPU);
- вбудована пам'ять програм: 8 ГБ EMMC Flash пам'ять;
- зовнішня пам'ять програм: TF карта (Макс. 64 ГБ);
- мережний інтерфейс: 10/100/1000 М Ethernet RJ45;
- бездротовий інтерфейс: WI-FI Realtek RTL8189ETV, IEEE 802.11 b/g/n;
- відео вхід: CSI роз'єм камери з підтримкою 8-бітних YUV422 CMOS камер CCIR656 протоколу для NTSC і PAL;
- дозвіл відео-захоплення: до 1080p @ 30fps;
- аудіо вхід: вбудований мікрофон;
- відео вихід: HDMI з HDCP і підтримкою HDMI CEC і інтегрований інтерфейс CVBS. Виходи можна використовувати одночасно;
- аудіо вихід: 3.5 мм роз'єм та HDMI;
- USB інтерфейс: три USB 2.0 HOST порту й один USB 2.0 OTG;

– низькорівнева периферія: 40 Pins GPIO інтерфейс, сумісний з Raspberry Pi B +.

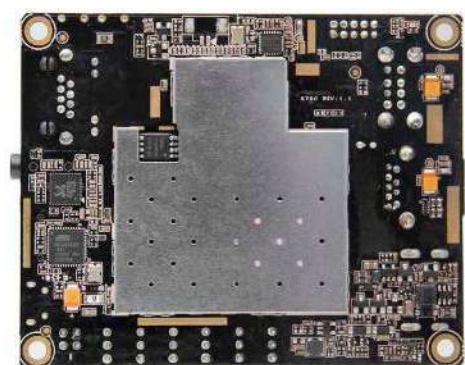
Підтримувані OS: Android, Lubuntu, Debian, Raspbian

2.5 LattePanda

LattePanda 2GB/32GB – це повнофункціональний одноплатний комп'ютер під керуванням операційної системи Windows 10 на 4-х ядерному процесорі Intel Cherry Trail Z8350 з обсягом оперативної пам'яті 2 ГБ і із внутрішньої еммс пам'яттю 32ГБ. Має всю необхідну периферію що й звичайний ПК і сполучимо практично з усіма розповсюдженими обладнаннями: принтери, джойстики, камери й багато чого іншого. Будь-які периферійні обладнання, які працюють на ПК, будуть працювати на Lattepanda [166]. Lattepanda поставляється з попередньо встановленим з повним виданням Windows 10 Home Edition, що дозволяє запускати потужні інструмент, такі як Visual Studio, Nodejs, Java, обробка й багато інших [167]. Lattepanda унікальний тим, що для зв'язку із зовнішнім миром комп'ютер містить інтегрований контролер Arduino Leonardo. Мінікомп'ютер знайде широке застосування в портативних системах, що вбудовуються, системах вилученого збору інформації й керування, 3D принтерах і т.д. Загальний вид одноплатного ПК Lattepanda 2GB/32GB представлено на рисунку 2.16.



а)



б)

Рисунок 2.16 – Загальний вид одноплатного ПК LattePanda 2GB/32GB [166]

Основні характеристики LattePanda 2GB/32GB:[68]

- процесор: Intel Cherry Trail Z8350 Quad Core;
- базова частота процесора: 1.44ГГц (1.92Ghz максимальна) ;
- операційна система: Windows 10 Home Edition (неактивована) ;
- оперативна пам'ять: 2GB DDR3L;
- еммс пам'ять: 32GB;
- графічний прискорювач Intel HD Graphics, 12 Eus @200-500Mhz;
- USB інтерфейси: USB 3.0 x 1, USB 2.0 x 2;
- бездротовий інтерфес: Wi-Fi 802.11n 2.4G;
- Bluetooth: 4.0;
- Arduino інтерфейс: Atmega32u4 (Arduino Leonardo) ;
- відео вихід: HDMI і MIPI-DSI;
- роз'єм підключення сенсорною панеллю;
- Ethernet: 100Mbps;
- GPIO входи/виходи;
- GPIO процесора Intel x 6;
- GPIO контролера Atmega32U4 x 20;
- інтерфейсні роз'єми x 6;
- напруга живлення: 5В 2А – рекомендуємо даний блок живленням;
- розміри плати: 88 x 70мм;
- вага: 55g.

2.6 Odroid

ODROID-XU4Q – це покоління обчислювальних обладнань із могутнішим, більш енергоефективним устаткуванням і меншим форм-фактором [168, 169]. Тепер це ще й самий тихий продуктивний міні-комп'ютер. Пропонуючи підтримку з відкритим вихідним кодом, плата може працювати з різними версіями Linux, включаючи останню версію Ubuntu

16.04 і Android 4.4 Kitkat, 5.0 Lollipop і 7.1 Nougat. Завдяки реалізації інтерфейсів EMMC 5.0, USB 3.0 і Gigabit Ethernet, ODROID-XU4 може похвалитися неймовірною швидкістю передачі даних – функцією, яка усе більше потрібна для підтримки поліпшеної обчислювальної потужності на обладнаннях ARM. Це дозволяє користувачам по-справжньому випробувати новинки в області обчислювальної техніки, особливо з більш швидким завантаженням, переглядом веб-сторінок, мережними й 3D-іграми. Дана модель є самою тихою й продуктивною у всій лінійці ODROID міні-комп'ютерів.

ODROID-XU4Q, оснащено чотирма продуктивними ядрами (ARM® Cortex® -A15™ до 2,0 ГГц) і чотирма невеликими ядрами (ARM® Cortex® -A7™ до 1,4 ГГц), забезпечує поліпшені можливості обробки при збереженні найбільш ефективної споживаної потужності. Завдяки розв'язку big.LITTLE™ HMP Exynos-5422 може використовувати максимум усі вісім ядер для керування обчислювальними завданнями. Графічний процесор ARM® Mali™-T628 MP6 пропонує повну підтримку API для OpenGL ES 1.1, OpenGL ES 2.0 і OpenGL ES 3.0, OpenCL 1.2 Full Profile і Google Renderscript. Mali-T628 – це кращий графічний процесор для використання в останніх поколіннях лідируючих на ринку обладнань, оптимізований для створення приголомшливих графічних ефектів для таких споживчих додатків, як 3D-Графіка, візуальні обчислення, доповнена реальність, генерація процедурних текстур і розпізнавання голосу.

Для завантаження ОС потрібна карта microSD або модуль EMMC. Рекомендуємо використовувати модуль emmc для більш швидкого завантаження ОС, більш швидкого запуску додатків, гладкої багатозадачності й ефективного доступу до хмари [69–70]. Загальний вид ODROID-XU4Q представлено на рисунку 2.17.



Рисунок 2.17 – Загальний вид ODROID-XU4Q [169]

Характеристики ODROID-XU4Q:

- процесори Samsung Exynos5422 Cortex™-A15 з тактовою частотою 2 ГГц і Cortex™-A7 Octa;
- Mali-T628 MP6 (повний профіль OpenGL ES 3.1 / 2.0 / 1.1 і OpenCL 1.2) ;
- 2 Гб оперативної пам'яті LPDDR3 ОЗУ (750 МГц, пропускна здатність пам'яті 12 ГБ / з, шина 2x32 біт);
- eMMC5.0 HS400 Flash Storage. Роз'єм для модуля eMMC: флеш-накопичувач eMMC 5.0 (до 64 ГБ) Слот для карти Microsd (до 128 ГБ)м
- 2 x USB 3.0 Host, 1 x USB 2.0 Host;
- порт Gigabit Ethernet 10/100/1000 Мбит/із із роз'ємом RJ-45 (підтримка Auto-mdix);
- Wi-Fi: IEEE 802.11 AC / B / G / N 1T1R WLAN з антеною (зовнішній Usb-Адаптер);
- HDMI 1.4a для підключення дисплея або телевізора;
- розмір: приблизно 83 x 58 x 20 мм (без кулера);
- живлення: 5 В / 4 А.

Підтримувані операційні системи:

- Ubuntu 16.04 + OpenGL ES + OpenCL в Linux Ядро 4.14 LTS;
- Android 4.4.2 у ядрі LTS 3.10;
- Android 7.1 доступна в якості розробки ОСм;
- розміри: 83 x 58 x 20 мм (вага: 38 грам) без кулера.

ODROID-XU4 – побудований на процесорі ARM® big.LITTLE™ technology, із застосуванням Heterogeneous Multi-Processing (HMP) технології. ODROID-XU4 нове покоління потужних, енергоефективних міні-комп'ютерів. Підтримується програмне забезпечення з відкритим вихідним кодом. На платі можна запустити різні клони Linux, включаючи Ubuntu 15.04 і Android 4.4 Kitkat і 5.0 Lollipop [170,171].

Вбудовані EMMC 5.0, USB 3.0 і Gigabit Ethernet інтерфейс. ODROID-XU4 має дивно велику швидкість передачі даних. Користувачі по гідності оціняють можливості комп'ютера при перегляді інтернет сторінок у мережних додатках і 3D іграх. Загальний вид ODROID-XU4 представлено на рис. 2.18.

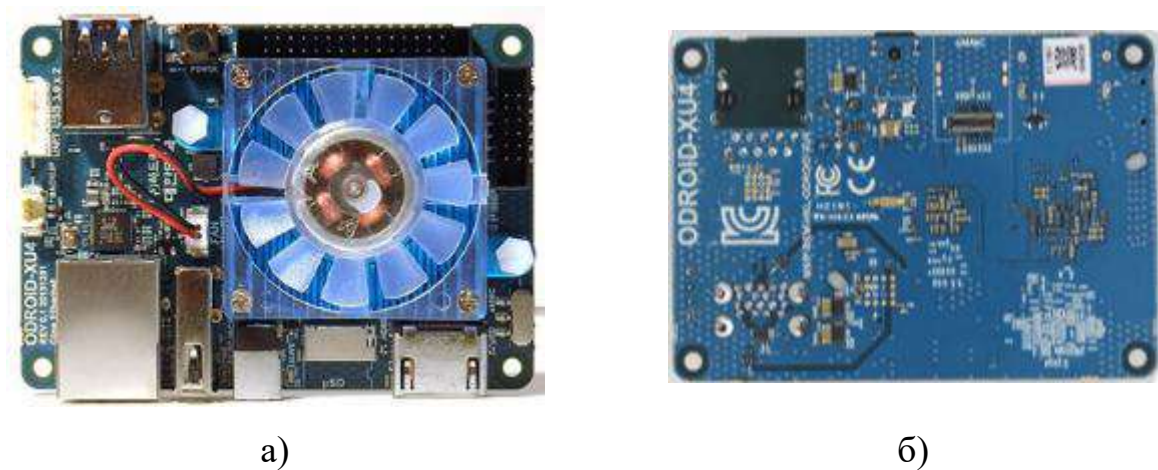


Рисунок 2.18 – Загальний вид ODROID-XU4 [170]

Основні характеристики ODROID-XU4:

- центральний процесор Samsung Exynos5422 Cortex™-A15 2Ghz і 8-мі ядерний процесор Cortex™-A7;
- Mali-T628 MP6(підтримка OpenGL ES 3.0/2.0/1.1 і OpenGL 1.1 повний профіль);
- 2 Гбайта LPDDR3 RAM PoP;
- флеш накопичувач EMMC5.0 HS400;
- 2 x USB 3.0 Host, 1 x USB 2.0 Host;
- гігабітний Ethernet порт;

- інтерфейс HDMI 1.4a;
- розмір: 82 x 58 x 22 мм (включаючи кулер процесора).

ODROID-HC2 – це 64-розрядна платформа на 4-ядерному процесорі Amlogic ARM® Cortex®-A53(Armv8), що працює на частоті 1,5 ГГц. Платформа призначена для побудови високопродуктивних систем автоматизації й обробки даних, домашніх кінотеатрів, прототипування обладнань, що вбудовуються [172,173]. Платформа ODROID-HC2 підтримує такі операційні системи як Ubuntu, Android, Archlinux, Debian і інші портовані 64-розрядні операційні системи з відкритим вихідним кодом. У комп'ютері сполучається мале споживання, розміри й висока продуктивність із можливістю підключення зовнішніх датчиків, обладнань і виконавчих механізмів. По багатьом параметрам ODROID-C2 більш ніж в 2 рази перевершує такий популярний міні-комп'ютер як Raspberry Pi 2 В. Загальний вид представлено на рисунку 2.19.

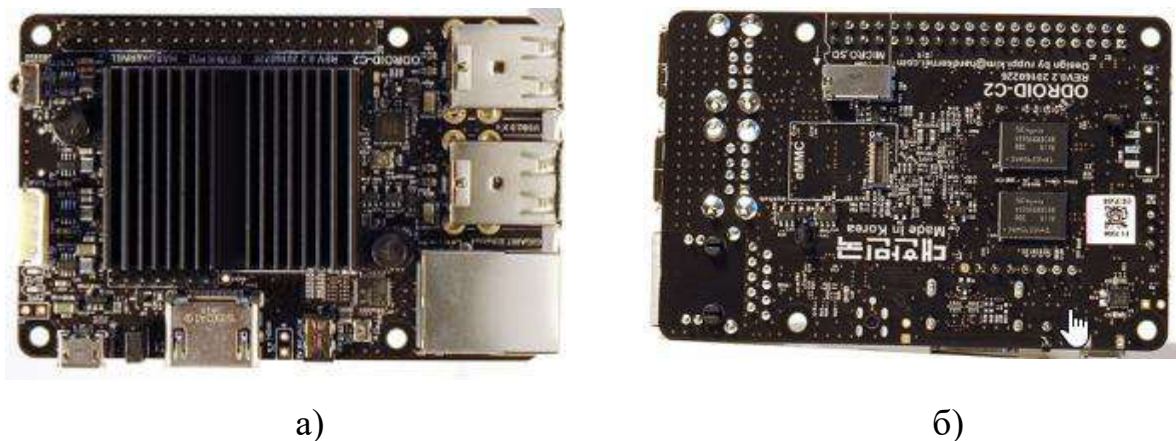


Рисунок 2.19 – Загальний вид ODROID-HC2

Основні характеристики ODROID-HC2:

- процесор: Amlogic ARM® Cortex®-A53(Armv8) 1,5 Ghz quad core;
- графічний процесор: Mali™-450 GPU (3 Pixel-processors + 2 Vertex shader processors);
- оперативна пам'ять: 2Gbyte DDR3 SDRAM;

- мережний інтерфейс: 10/100/1000Mbps Ethernet with RJ-45;
- бездротовий інтерфейс: USB IEEE 802.11b/g/n WLAN (опційно);
- відео вихід: HDMI 2.0 4K/60Hz;
- вхід камери: USB 720p (опційно);
- підтримувані формати відео: H.265 4K/60FPS and H.264 4K/30FPS capable VPU;
- порти вводу/вивіду: 40pin Gpios + 7pin I2S;
- зберігання програм і даних: emmc5.0 HS400 і UHS-1 SDR50 microSD;
- порти USB: USB 2.0 Host x 4, USB OTG x 1 (power + data capable) ;
- додаткові порти: інфрачервоний порт;
- операційні системи: Ubuntu 16.04 або Android 5.1 Lollipop based on Kernel 3.14LTS;
- розміри: 85 x 56 x 18 мм;
- вага: 56 з радіатором.

2.7 PINE64

PINE A64+ Board 1Гб – 1,2 ГГц 64- розрядному чотириядерному процесорі ARM Cortex A53. Комп'ютер обладнано оперативною пам'яттю DDR3 обсягом 1Гбайт [174]. Графічна підсистема посилена двоядерним графічним співпроцесором Mali 400 MP2, що дозволяє без проблем переглядати відео з дозволом до 4К.

На платі розміщено два GPIO роз'єми – один Raspberry Pi2 сумісний, другий формату Euler Bus. Наявність такої кількості цифрових портів робить Pine64 гідним конкурентом Raspberry для систем, що вбудовуються, керування, де потрібна велика кількість GPIO входів/виходів. Передбачені роз'єми для підключення батарейного живлення (є перемички, що перемикають живлення комп'ютера як від зовнішніх 5В так і від батарей 3.6В) комп'ютера й резервного живлення годин реального часу. До плати

можна підключити дисплей із цифровою шиною, відеокамеру й сенсорну панель. Загальний вид PINE A64+ Board 1Гб представлено на рисунку 2.20

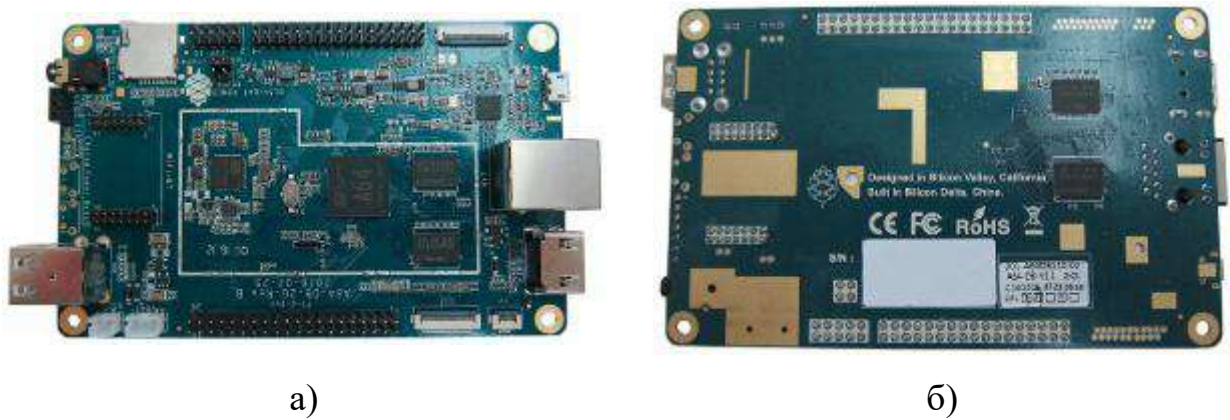
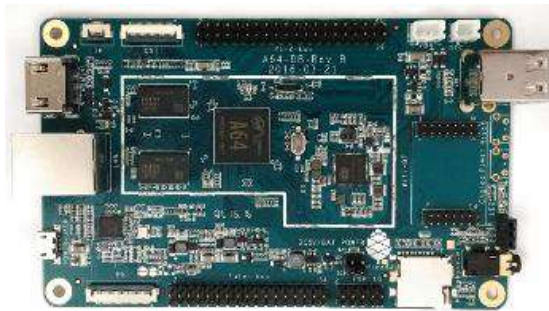


Рисунок 2.20 – PINE A64+ Board 1Гб [175]

Основні характеристики PINE A64+ Board 1Гб:

- процесор: 1.2 ГГц Quad-Core ARM Cortex A53 64-біт процесор з виконанням як 32 так 64-бітних команд;
- цифрове введення/вивід: два роз'єми Pi2Bus і Euler Bus;
- графічний процесор: двоядерний Mali 400 MP2;
- оперативна пам'ять: 1 Гбайт;
- графічна підсистема: інтегрована з HDMI 1.4 виходом з якістю до 4K
- мережне обладнання: 10/100/1000 Ethernet порт;
- бездротове обладнання: опційно Bluetooth 4.0 і 802.11BGN бездротовий модуль.

PINE A64-LTS – комп'ютер на потужному 64-бітному процесорі Quad-Core ARM Cortex A53 від Allwinner «R18». Комп'ютер оснащений оперативною пам'яттю DDR3 обсягом 2Гбайта й завантажувальною Flash-Пам'яттю 128 МБ SPI. Графічна підсистема посилена двухядерним графічним співпроцесором Mali 400 MP2, що дозволяє без проблем переглядати відео з дозволом до 4K [176]. Загальний вид міні комп'ютера PINE A64-LTS представлено на рисунку 2.21.



а)



б)

Рисунок 2.21 – Загальний вид PINE A64-LTS [177]

Основні характеристики PINE A64-LTS представлені нижче [177]:

- процесор: 64-бітного процесора Quad-Core ARM Cortex A53 від Allwinner «R18» процесор з виконанням як 32 так 64-бітних команд;
- цифрове введення/вивід: два роз'єми Pi2Bus і Euler Bus;
- графічний процесор: двухядерний Mali 400 MP2;
- оперативна пам'ять: 2 Гбайт;
- графічна підсистема: інтегрована з HDMI 1.4 виходом з якістю до 4K;
- мережне обладнання: 10/100/1000 Ethernet порт;
- бездротове обладнання: опційно Bluetooth 4.0 і 802.11BGN бездротовий модуль;
- підтримка Android, Linux, Remix OS, Windows 10 IoT;
- живлення: 5V 2A (роз'єм 3.5x1.35мм, але підходить і 4.0x1.7мм);
- розміри: 133 x 80 x 19mm.

Інтерфейси:

- 2 порту USB 2.0;
- роз'єм HDMI 1.4a;
- роз'єм 3.5 jack;
- 3 x-контактний роз'єм для підключення літієвої батареї;
- 2 x-контактний роз'єм для підключення RTC батареї;
- роз'єм для підключення камери, дисплея й тачскрина;

- вбудований мікрофон;
- вбудований ІК-приймач;
- 3 набору GPIO пінів: 2x20 RPi пінів, 2x17 Euler пінів і 2x5 EXP;
- роз'єм для підключення Wi-Fi або Bluetooth модуля.

2.8. Модуль ESP32-CAM

ЕІ ESP32-CAM – це модуль, який можна використовувати з безліччю проектів та Arduino. Це закінчений модуль із вбудованим мікроконтролером, завдяки якому він може працювати незалежно. На додаток до підключення Wi-Fi + Bluetooth цей модуль також має вбудовану відеокамеру і слот microSD для зберігання [178].

Цей модуль зовсім не дорогий і може мати багато додатків. Від простого IoT до іншого, більш просунутого – для моніторингу та розпізнавання зображень з використанням ШІ, і навіть як система спостереження для віддаленої перевірки того, що відбувається в будь-якому місці, де б ви не знаходилися [179]. Загальний вигляд ESP32-CAM без програматора представлений на рис. 2.22.



Рисунок 2.22 – Загальний вигляд ESP32-CAM без програматора [179]

Модуль ESP32-CAM містить таке компонування:

- бездротовий модуль ESP32-S з інтегрованим Wi-Fi та Bluetooth контролерами;

- камера OV2640 – йде окремо, але на платі передбачено роз'єм для її підключення;

- гніздо для карт пам'яті micro-SD;

- світлодіод, який можна використовувати у вигляді спалаху або підсвічування, залежно від підставлених завдань.

ESP32-CAM має деякі технічні характеристики:

- Wi-Fi 802.11b/g/n+Bluetooth 4.2 з BLE. Підтримує завантаження зображень через Wi-Fi;

- UART, SPI, I2S ШІМ. Він має 9 контактів GPIO;

- тактова частота: до 160 МГц;

- обчислювальна потужність мікроконтролера: до 600 DMIPS;

- пам'ять: 520 КБ SRAM + 4 МБ PSRAM + слот для SD-карти;

- екстракт: має кілька сплячих режимів, прошивку з можливістю оновлення по OTG та світлодіоди для використання вбудованої флеш-пам'яті;

- камера: підтримує камери OV2640, які можна придбати у комплекті або придбати окремо. Ці типи камер мають: 2 МП на вашому сенсорі; розмір масиву UXGA 1622 × 1200 пікселів; формат виведення YUV422, YUV420, RGB565, RGB555 та 8-бітне стиснення даних. Можливість передавати зображення від 15 до 60 кадрів за секунду.

Розпіновка виводів El ESP32-CAM представлено рисунку 2.23.

Підключення для прошивки El ESP32-CAM за допомогою ESP32-CAM-MB нескладне, а ось підключення для прошивки за допомогою SB-UART є відповідні вимоги до перехідника на базі FTDI, якому необхідно забезпечити таке з'єднання:

- підключіть 5-вольтове з'єднання модуля ESP32-CAM до Vcc модуля FTDI;

- підключіть GND модуля ESP32-CAM до GND модуля FTDI;

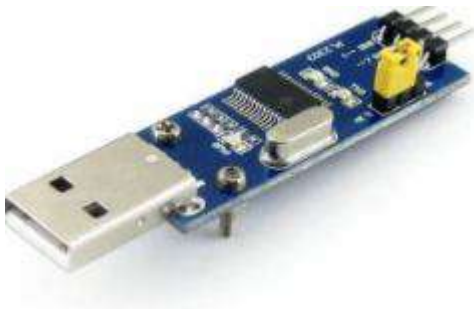
- TX0 із плати FTDI переходить у GPIO 3 (U0RXD);

- RXI із плати FTDI йде на GPIO 1 (U0TXD);
- обходить GPIO 0 та GND плати ESP32-CAM.



Рисунок 2.23 – Розпіновка виводів EI ESP32-CAM

Прошивка EI ESP32-CAM проводиться через UART [180], тому можуть використовуватися два види USB-UART перехідників (рис. 2.24).



а)

а) SB-UART перехідник на базі PL2303;



б)

б) ESP32-CAM-MB

Рисунок 2.24 – USB-UART перехідники для прошивки ESP32-CAM [181]

Схема підключення представлена рисунку 2.25.

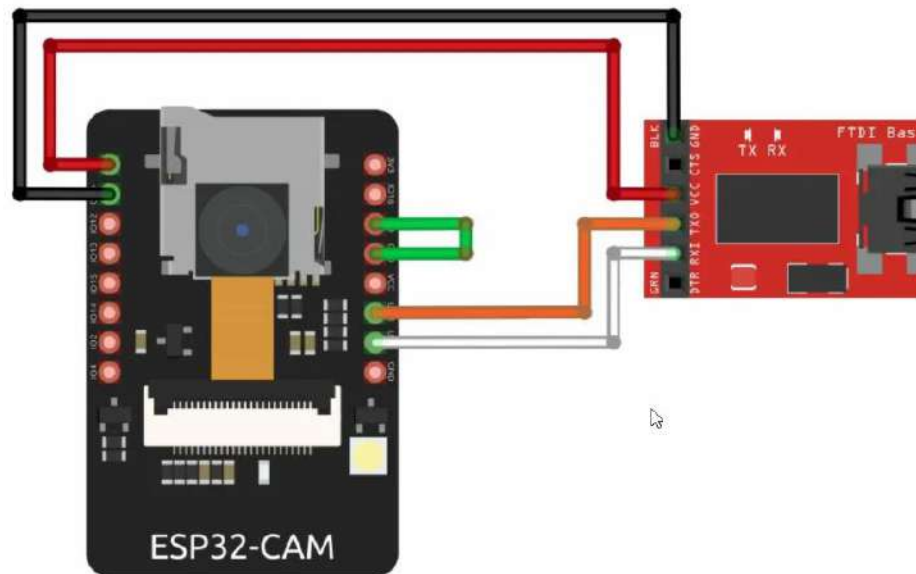


Рисунок 2.25 – Підключення ESP32-CAM та FTDI для прошивки

При підключенні для прошивки за допомогою перехідника на базі PL2303 (рисунок 2.24, а) використовуються виводи:

- GPIO1 – U0TXD – підключається до Rx UART;
- GPIO3 – U0RXD – підключається до Tx UART.

Крім того, для програмування модуля необхідно підтягнути виводи GPIO0 на землю. Відповідно, після закінчення прошивки, підтяжку треба прибрати. Причому при підключенні живлення через вхід 3.3В спостерігалася велика кількість різноманітних проблем – прошивка відбувається через раз, модуль не спрацьовував і ці проблеми разом усунулися при подачі 5В живлення через відповідний вхід. Варто звернути увагу, що напруга логічних рівнів на входах – суворо 3.3 В. Сигнали 5 В можуть пошкодити контролер.

Налаштування Arduino IDE для прошивки ESP32-CAM.

Крок 1. Установка Arduino IDE з офіційного сайту:
<https://www.arduino.cc/en/Main/Software> [182]

Крок 2. Потрібно інтегрувати підтримку нашого модуля ESP32. Ідемо в налаштування (File – Preferences) і в нижній частині вікна налаштувань в Additional Boards Manager URLs додаємо рядок:

https://dl.espressif.com/dl/package_esp32_index.json
http://arduino.esp8266.com/stable/package_esp8266com_index.json

Отриманий результат повинен виглядати так, як представлено рисунку 2.26.

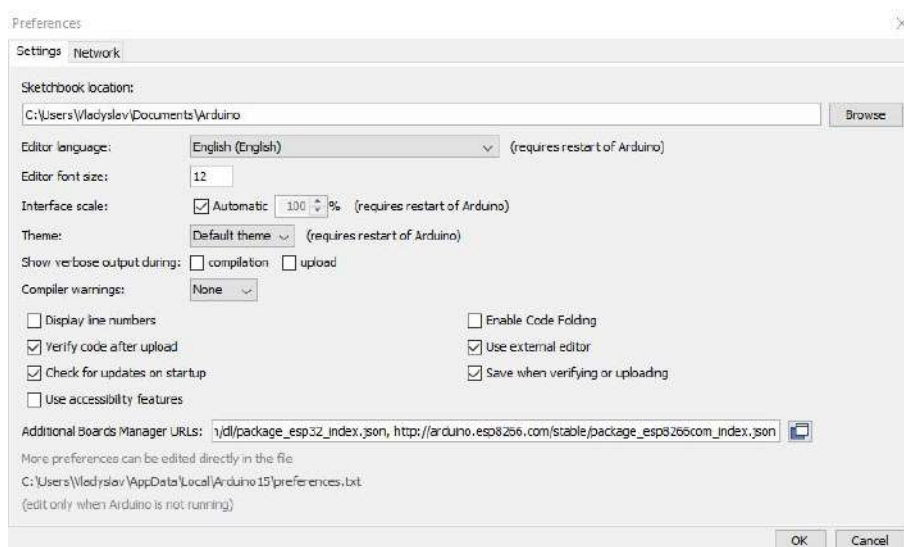


Рисунок 2.26. – Налаштування підтримки модулів ESP32.

Крок 3. Переходимо до Boards Manager: Tools – Board – Boards Manager для встановлення пакету ESP32, як представлено рисунку 2.27.

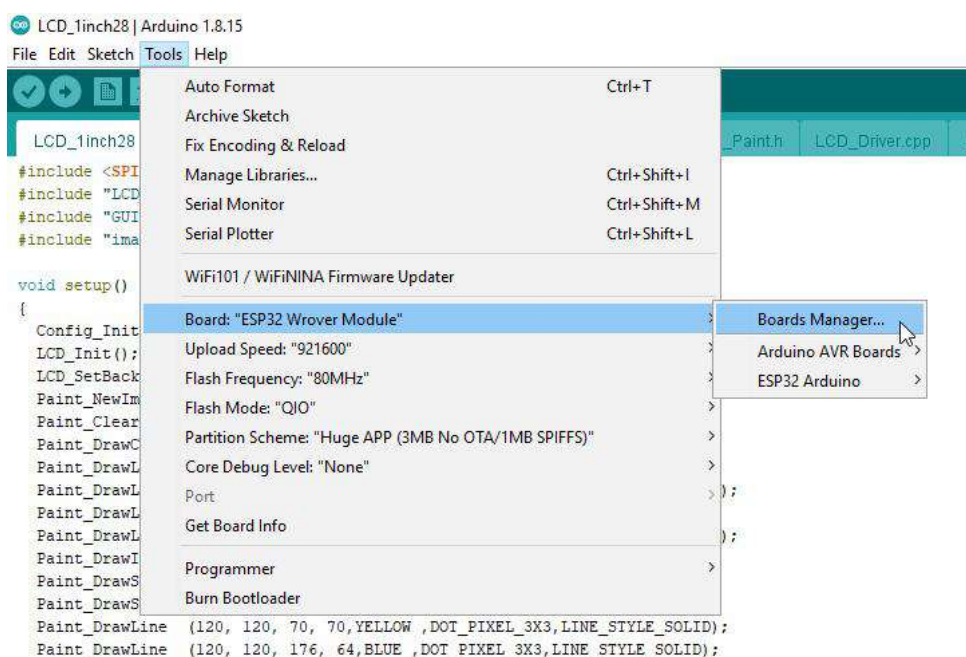


Рисунок 2.27 – Виклик вікна Boards Manager

У вікні Boards Manager у рядку пошуку Type введіть "esp32" і встановлюємо (Install) пакета ESP32 by Espressif Systems, як показано на рисунку 2.28.

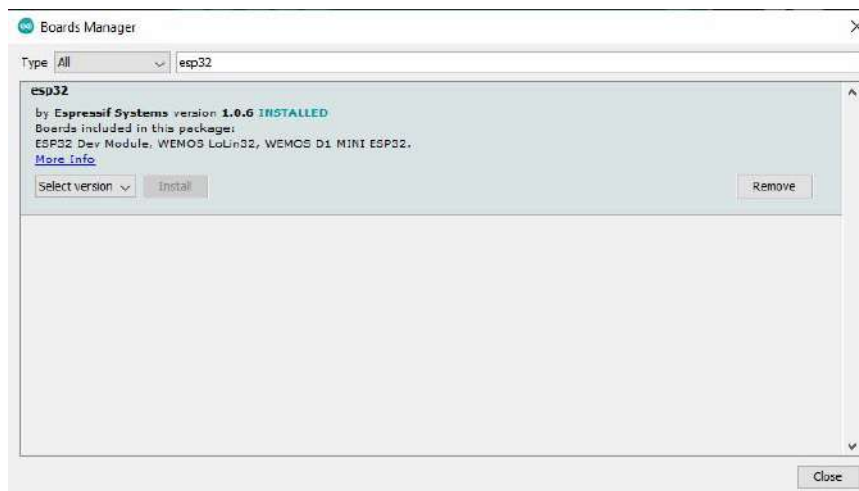


Рисунок 2.28 – Вікно завантаження та інсталяції пакета ESP32 by Espressif Systems

Тестування працездатності ESP32-CAM на базі Examples. Для перевірки працездатності ESP32-CAM і правильності підключення USB-UART перехідників для прошивки необхідно виконати такі кроки:

Крок 1. Відкриваємо приклад Arduino IDE для роботи з камерою. Для цього слідуємо у File – Examples – ESP32 – Camera – CameraWebServer. Приклад шляху представлено на рис. 2.29.

Крок 2. Проводимо налаштування модуля камери. Для цього необхідно розкоментувати наступний рядок і закоментувати всі інші. У файлі CameraWebServer коментувати:

```
#define CAMERA_MODEL_AI_THINKER // Has PSRAM
```

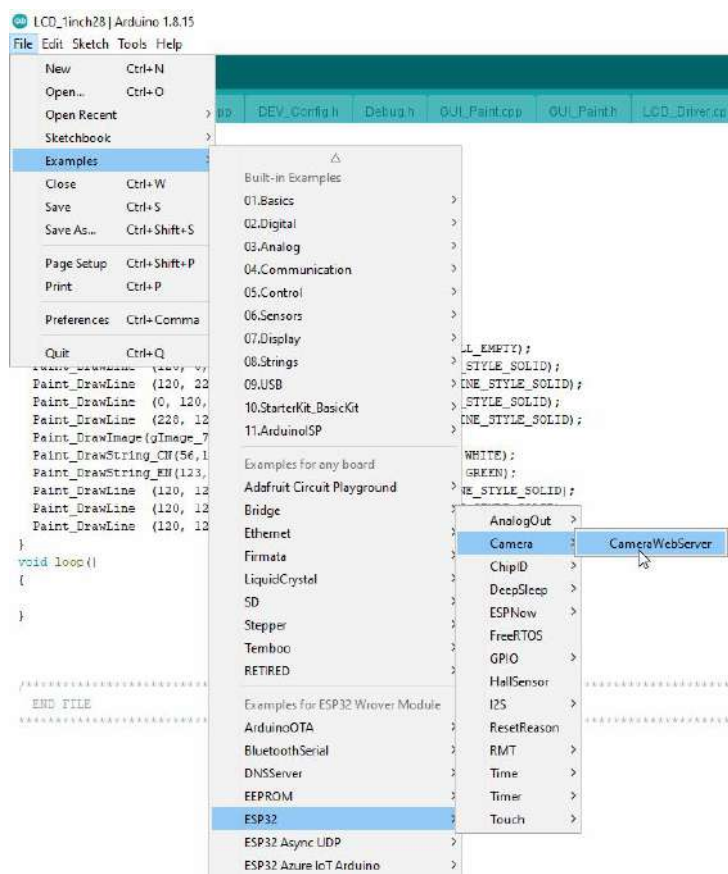


Рисунок 2.29 – Завантаження тестового скетчу для ESP32-CAM

Крок 3. Необхідно вказати SSID і password Wi-Fi мережі, в якій працюватиме ESP32-CAM, для цього необхідно внести дані в рядках:

```
const char* ssid = "*****";
const char* password = "*****";
```

* нагадування SSID – назва мережі; password – пароль підключення.

Крок 4. У меню Tools потрібно встановити актуальні параметри для прошивки плати ESP32-CAM. Для цього необхідно обрати плату, номер COM порту для прошивки. Приклад параметрів підключення наведено на рисунку 2.30.

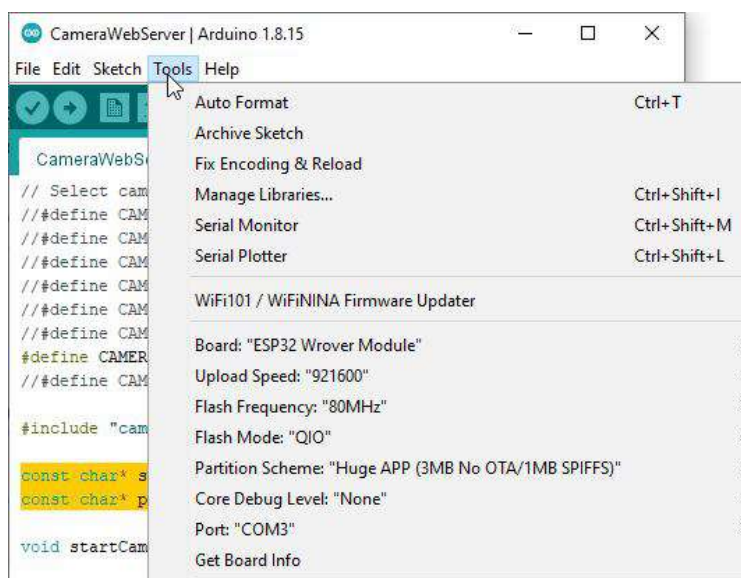


Рисунок 2.30 – Параметри для прошивки плати ESP32-CAM

Крок 5. Прошиваємо плату ESP32-CAM. Для цього натискаємо кнопку UPLOAD, не забувши попередньо підтягнути пін GPIO0 до землі, якщо використовується для прошивки USB-UART перехідник на базі PL2303 (рис. 2.31).



Рисунок 2.31 – Прошивка ESP32-CAM

Після завершення прошивки ESP32-CAM у вікні Arduino IDE з'являється наступний запис

Leaving...

Hard resetting via RTS pin...

Це означає, що необхідно перезавантаження ESP32-CAM і при цьому необхідно відключити підтяжку GPIO0.

Крок 6. Проведемо перевірку правильності прошивки та підключення до Wi-Fi мережі. І тому заходимо в Tools-Serial monitor, як показано рис. 2.32.

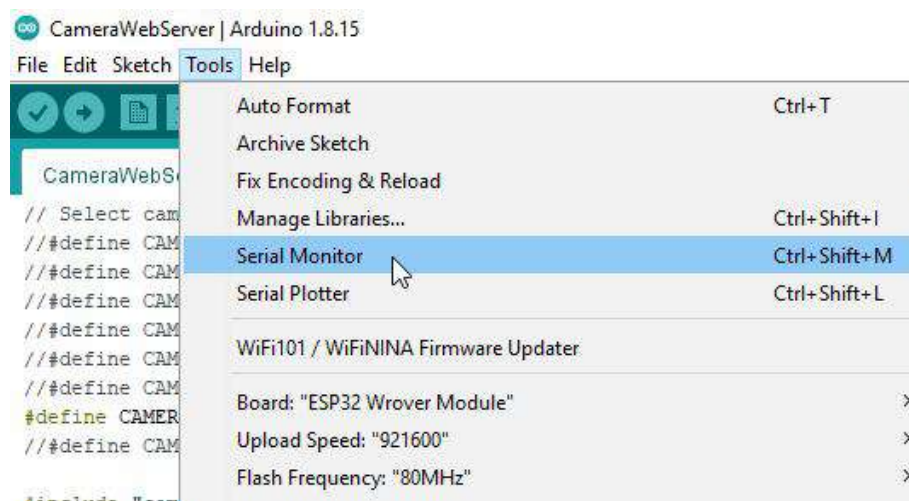


Рисунок 2.33 – Шлях для вибору Serial Monitor

При успішному з'єднанні з поточною Wi-Fi мережею в моніторі порту буде наступний запис, який представлений на рисунку 2.34.

```
.....
WiFi connected
Starting web server on port: '80'
Starting stream server on port: '81'
Camera Ready! Use 'http://192.168.43.1' to connect
```

Рисунок 2.34 – Вікно монітора порту

Для перевірки працездатності ESP32-CAM необхідно зайти на ПК або на мобільному телефоні, підключеному в цю Wi-Fi мережу за адресою <http://192.168.43.1>

Крок 7. При переході на адресу присвоєному модулю ESP32-CAM вашою Wi-Fi мережею, стане доступний інтерфейс налаштування та запуску трансляції, загальний вигляд якого представлений на рисунку 2.35.

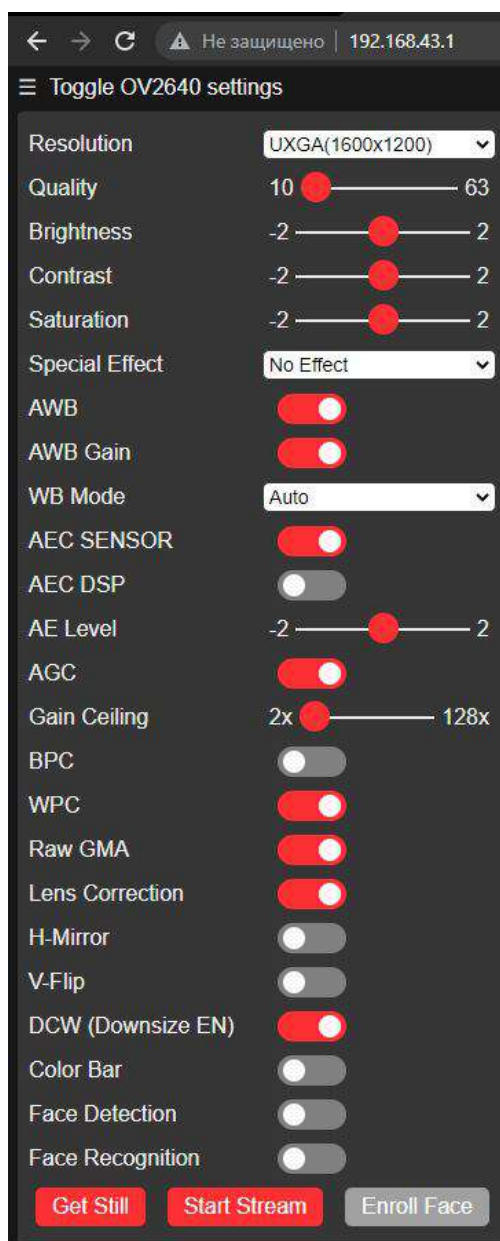


Рисунок 2.35 – Інтерфейс налаштування трансляції ESP32-CAM

Щоб почати транслювати потокове відео з модуля ESP32-CAM, натисніть Start Stream. Приклад отриманого результату трансляції відео в режимі UXGA з роздільною здатністю 1600x1200 представлений на рис. 2.36.

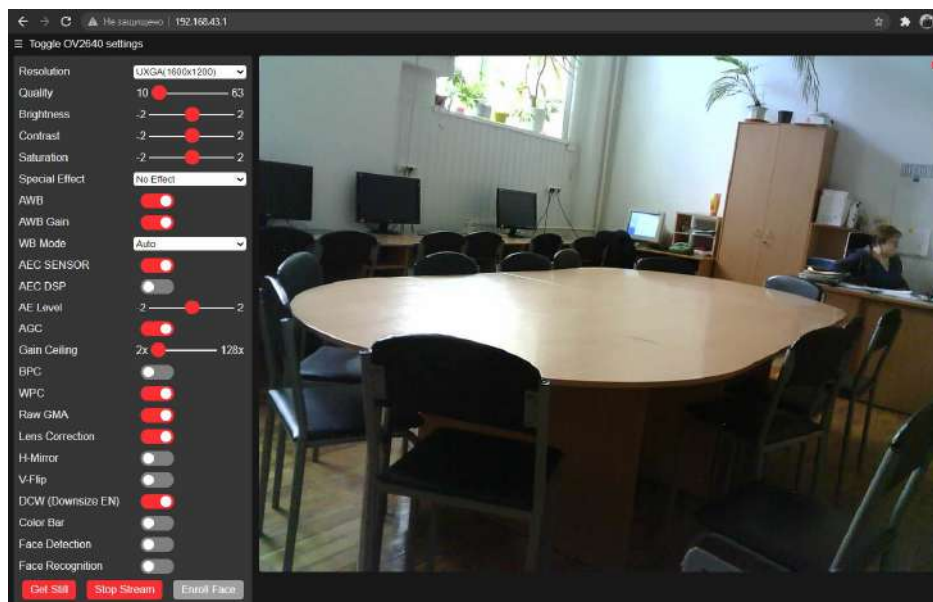


Рисунок 2.36. – Трансляція відео в режимі UXGA з роздільною здатністю 1600x1200

2.9 Плати розширень, модулі

Arduino Expansion Shield om DFRobot (www.DFRobot.com)

Для Arduino немає нічого простіше, чим управляти двигунами з реакцією на датчики перешкоди. Тепер переваги обох обладнань об'єднані в новому shield для Raspberry Pi. На shield виведені контакти введення/вивід в стандартному форм-факторі Arduino, що дозволяє використовувати готові шилди й плати для контролера Arduino. Додатково виведені контакти GPIO Raspberry, що дозволяє максимально використовувати ресурси як Raspberry Pi так і Arduino [183].

Shield підключається до Raspberri Pi за допомогою GPIO роз'єму й USB перехідника й дозволяє використовувати Arduino у якості віртуального послідовного порту, віртуальної клавіатури або віртуальної миші. На платі shield так само встановлений годинник реального часу для Raspberry на мікросхемі DS1307 з автономним живленням [184]. Загальний вид Arduino Expansion Shield від Dfrobot представлено на рисунку 2.37.

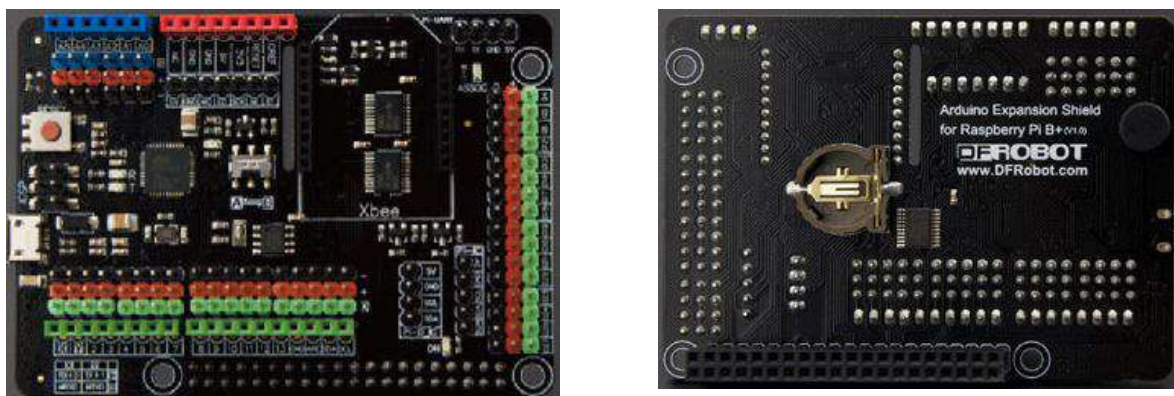


Рисунок 3.37 – Загальний вид Arduino Expansion Shield от DFRobot [184]

Основні характеристики Arduino Expansion Shield от DFRobot:

- бортовий мікроконтролер Atmega32u4;
- Arduino-сумісне розміщення контактів;
- сполучимо з усіма стандартними екранами й датчиками Arduino;
- підтримка стандарту інтерфейсів Gravity для UART і I²C інтерфейсів;
- напруга живлення Arduino 5 В;
- цифрових входів/виходів Arduino 20;
- аналогових входів Arduino 6;
- інтерфейси Raspberry Pi: Raspberry Pi B + GPIO(16); Raspberry Pi B + I2C(1); Raspberry Pi B + ID_I2C(1); Raspberry Pi B + SPI(1); Raspberry Pi B + TTL UART(1);
- розмір: 86 мм x 61 мм x 26 мм.

Роз'єм ICSP призначений тільки для внутрішньосхемного програмування мікроконтролера й не має доступу як Arduino введення/виводу. Перемикач на платі дозволяє підключити Xbee модуль як до Raspberry Pi так і до контролера Arduino.

Розташування й призначення висновків на Arduino Expansion Shield від Dfrobot представлено на рисунку 2.38.

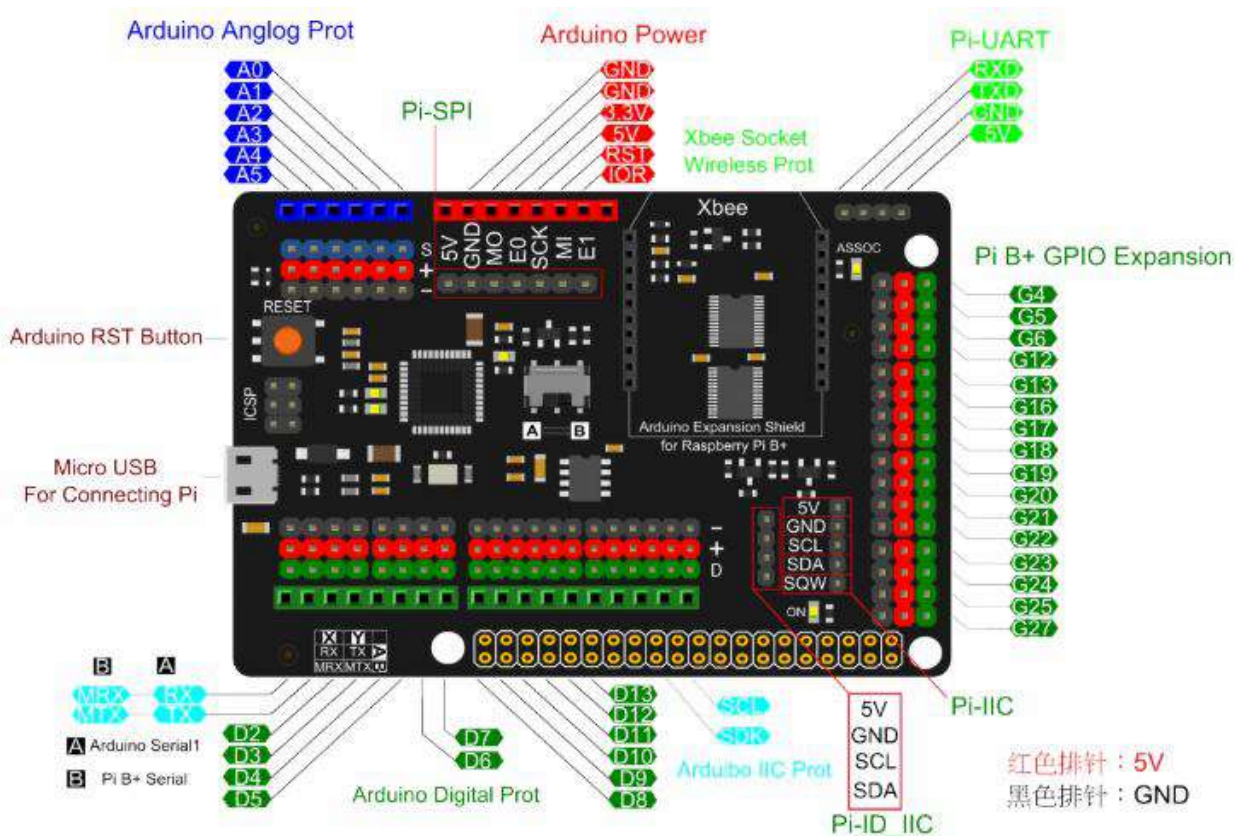


Рисунок 2.38— Розташування й призначення виводів на Arduino Expansion Shield

I²C модуль розширення на MCP23017

Модуль I²C адаптера графічного дисплея на мікросхемі MCP23017 призначений для спрощення підключення графічного дисплея 128x64 крапки (рекомендується на чипі KS0108) з паралельним інтерфейсом до контролера Arduino або міні-комп'ютеру Raspberry за допомогою послідовного I²C інтерфейсу. Модуль так само можна використовувати як звичайний розширник квазідвонаправлених портів введення-вивід [185].

I²C адреса обладнання вибирається перемичками на платі. За замовчуванням виставлена адреса 0x20, яку, при необхідності, можна змінити на кожній з діапазону 0x20-0x27. Дана операція вимагає застосування паяльника й навичок монтажу.

Вмикання й вимикання підсвічування контролюється так само через шину I²C. Якщо Вам потрібно управляти яскравістю підсвічування мікроконтролером, необхідно перерізати перемичку Backlight control на

платі й подати керуючий ШІМ сигнал на вивід LCD+ роз'єм [80]. Загальний вид модуля представлено на рисунку 2.39



Рисунок 2.39 – Загальний вид модуля I2C адаптера графічного дисплея [186]

Основні характеристики I2C модуль розширення на MCP23017:

– напруга живлення від 1,8В до 5,5В (при напрузі живлення нижче 4,5В знижується яскравість підсвічування й контрастність дисплея);

– максимальний споживаний струм 150 мА;

– інтерфейс I²C;

– режими роботи інтерфейсу 100кГц/400кГц/1,7МГц;

– I2C адреса обладнання від 0x20 до 0x27;

– I2C адреса за замовчуванням 0x20;

– розміри плати: 19x51 мм.

Швидкість роботи I²C модуль розширення на MCP23017:

– очищення екрана: 600 мс (тобто 0.6 секунди);

– установка курсору: 33 мс;

– друк тексту: приблизно 4 мс/символ;

– малювання графіки: приблизно 2 мс/байт;

– очищення прямокутної області: приблизно 120 мс для області розміром 20 x 50 крапок з вирівнюванням по 8 крапок по вертикалі;

– заповнення прямокутної області 20 x 50 крапок: 5230 мс;

– малювання рамки прямокутника в одну крапку 20 x 50 крапок: 730

мс;

– малювання рамки прямокутника у дві крапки 20 x 50 pixels: 1430 мс;

Розташування й призначення виводів на I²C модуль розширення на MCP23017 від Robotdyn представлено на рисунку 2.40.

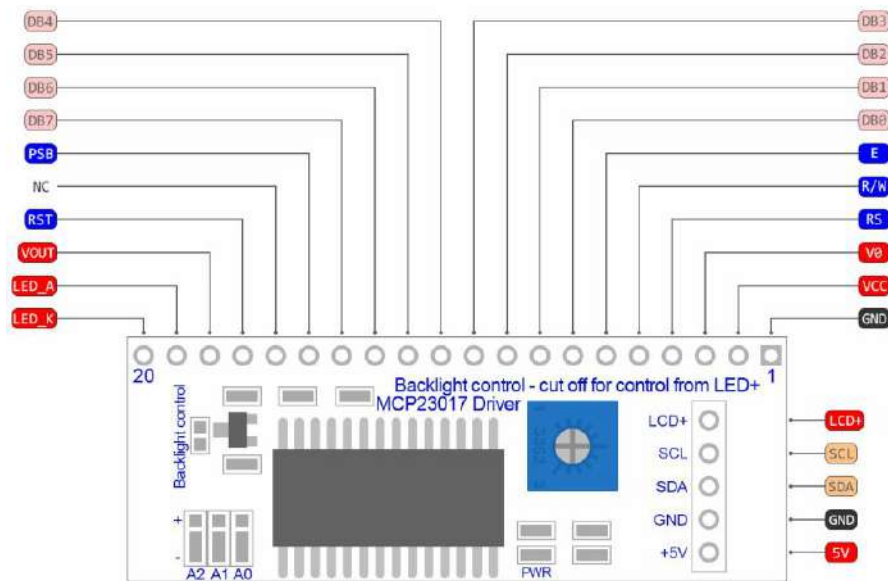


Рисунок 2.40 – Розташування й призначення виводів на I²C модулі розширення на MCP23017

З виводом тексту, що повільно змінюється або статичного, I²C модуль прекрасно справляється. Для високошвидкісної роботи із графікою необхідне тільки підключення через паралельний і SPI інтерфейс.

Модуль PoE живлення POE_BOARD для Raspberry Pi 3 B+

Модуль PoE живлення для міні-комп'ютера Raspberry Pi 3 B+ дозволяє заживлювати через Ethernet кабель, що дуже актуально для місць де є кабель, але немає можливості додатково підключити або встановити блок живлення. Плата відрізняється малими розмірами, високою якістю виготовлення й наявністю додаткового керованого вентилятора для примусового охолодження компонентів комп'ютера. Модуль комплектується елементами кріплення до плати міні-комп'ютера [187]. Загальний вид модуля PoE живлення POE_BOARD представлено на рисунку 2.41.

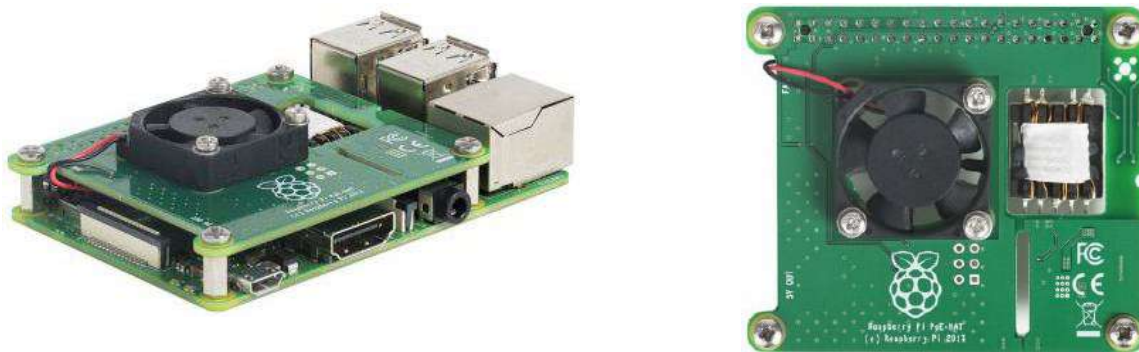


Рисунок 2.41 – Загальний вид модуля PoE живлення POE_BOARD [187]

Основні характеристики модуля PoE:

- сумісність із Raspberry Pi 3 B+;
- стандарт 802.3 af PoE;
- вхідна напруга від 36В до 56В;
- вихідна напруга 5В;
- максимальний вихідний струм 2,5А;
- керування охолодженням: мікроконтролерне;

Послідовність підключення модуля PoE до Raspberry Pi:

- підключити Raspberry Pi PoE HAT до контактів GPIO і PoE (40 + 4), які встановлені на Raspberry Pi.

Після підключення, плата PoE HAT готова до використання й Raspberry Pi може житися через мережний роз'єм RJ45.

Модуль живлення Li-ion 14500 5В для Raspberry Pi

Модуль живлення для міні-комп'ютерів сімейства Raspberry Pi побудований на мікросхемі керування електроживленням SW6106, забезпечує вихідну напругу живлення 5 В від Вашого акумулятора 14500, що робить Raspberry Pi автономним портативним обладнанням. Він також заряджає акумулятор, підтримує двонаправлену швидку зарядку. Цей модуль може використовуватися як універсальний міні-блок живлення й джерело безперебійного живлення, а також працює з іншими обладнаннями 5В [188].

Загальний вид модуль живлення Li-ion 14500 5В для Raspberry Pi представлений на рисунку 2.42.

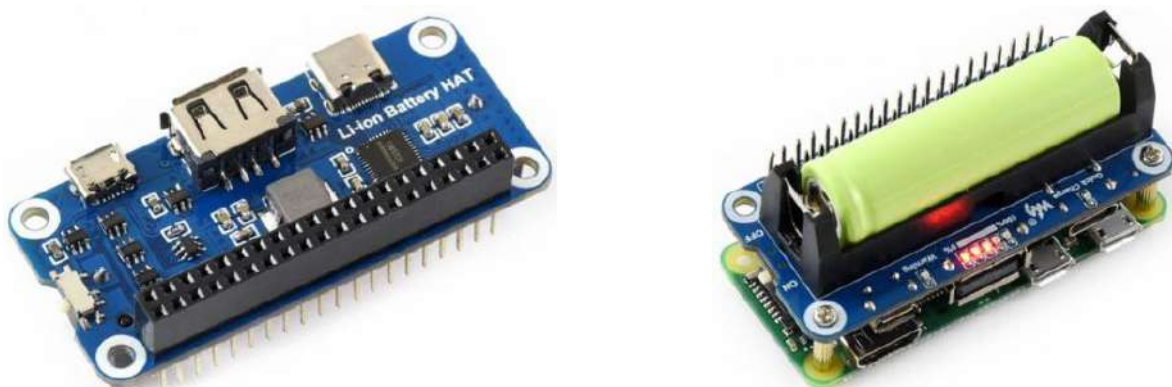


Рисунок 2.42 – Модуль живлення Li-ion 14500 5В для Raspberry Pi [188]

Основні характеристики модуля живлення Li-ion 14500 5В для Raspberry Pi:

- тип батареї Li-іо;
- типорозмір 14500;
- напруга батареї 3.7В (4.2У при повній зарядці) ;
- мікросхема перетворювача SW6106;
- вихідна напруга 5 В;
- роз'єм для Raspberry Pi GPIO;
- мікросхема керування живленням SW6106;
- роз'єм USB Type-C заряд батареї / вихідна напруга;
- роз'єм USB Type-A вихідна напруга;
- роз'єм Micro USB заряд батареї;
- вимикач живлення Raspberry Pi.

Додаткові функції модуля живлення Li-ion 14500 5В для Raspberry Pi:

- вбудований чип SW6106 підтримує двонаправлену швидку зарядку, підтримує протоколи з декількома швидкими зарядами, такі як PD/QC/FCP/PE/SFCP;

– вбудований захист акумулятора від зворотної полярності, захист від перезаряду / перерозряду, захист від перевантаження по струму й захист від короткого замикання;

– вбудовані індикатори: швидкої зарядки, інформація про ємність акумулятора й неправильної полярності акумулятора.

Модуль розширення DVK512 для Raspberry Pi

Плата розширення DVK512 від Waveshare – це плата, розроблена для лінійки міні-комп'ютерів Raspberry Pi, що поєднує в собі різні компоненти й інтерфейси для підключення зовнішніх аксесуарів [189]. Загальний вид модуль розширення DVK512 представлений на рисунку 2.43.

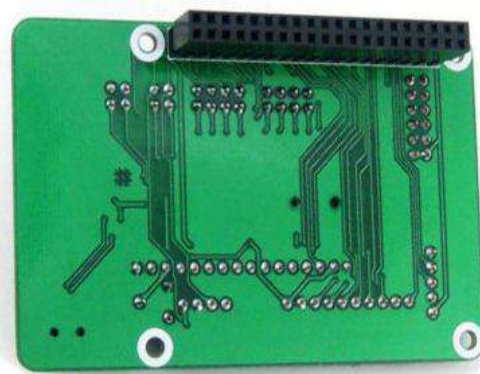


Рисунок 2.43 – Загальний вид модуль розширення DVK512 [189]

Модуль розширення DVK512 сполучимо з наступними моделями Raspberry Pi:

- Raspberry Pi 1 Model A+;
- Raspberry Pi 1 Model B+;
- Raspberry Pi 2 Model B;
- Raspberry Pi 3 Model B;
- Raspberry Pi 3 Model B+.

Розміщення компонентів на платі модуля розширення DVK512 представлено на рисунку 2.44.

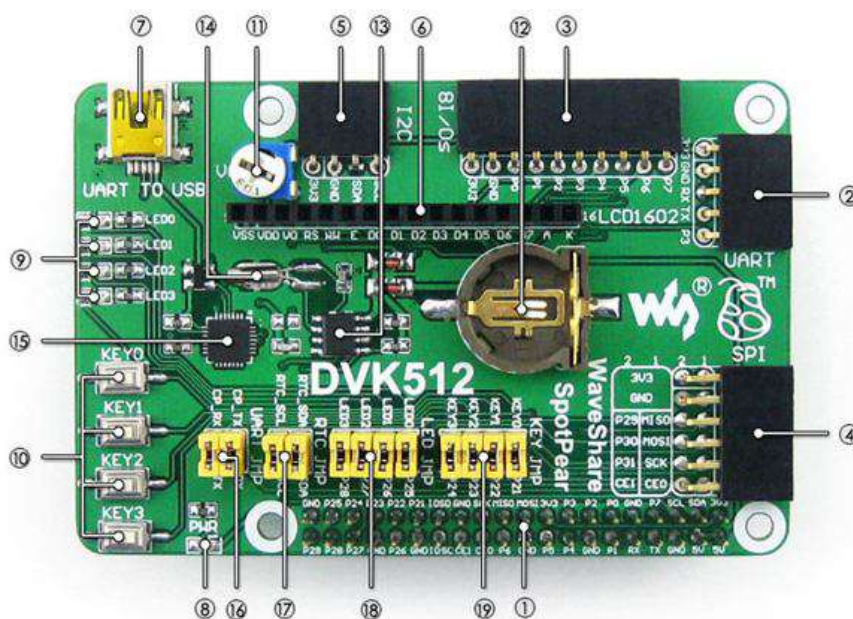


Рисунок 2.44 – Розміщення компонентів на платі модуля розширення DVK512 [189]

1. Роз'єм для підключення до Rpi;
2. Інтерфейс UART: підключається до модулів UART, таким як RS232, RS485, USB-UART і т.д. ;
3. Інтерфейс I²C:: з'єднується з модулями, керованими входами/виходами, такими як 8 кнопок Push, Logic Level Converter, Mix Board і т.д. ;
4. Інтерфейс SPI: підключається до Spi-модулів, таких як AT45DBXX Dataflash, плата L3G4200D і т.д.;
5. Інтерфейс I²C: підключається до модулів I²C, таких як плата розширення PCF8574, плата PCF8563 RTC, плата LSM303DLHC і т.д.;
6. Інтерфейс РК-дисплея: для підключення РК-дисплеїв, таких як LCD1602;
7. USB-роз'єм: USB-UART, підтримуваний вбудованим перетворювачем CP2102;
8. Індикатор живлення;
9. Світлодіоди призначені для користувача;
10. Призначені для користувача кнопки;

11. Потенціометр: для регулювання контрастності LCD1602;
12. Наявність батареї RTC;
13. Мікросхема PCF8563;
14. Кварцовий резонатор 32.768K годин реального часу;
15. Мікросхема перехідника USB-UART CP2102;
16. Перемичка включення CP2102v
17. Перемичка включення годин реального часу;
18. Перемичка світлодіодів користувача;
19. Перемички для кнопок користувача.

Модуль 3-х реле для Raspberry Pi

Модуль реле для Raspberry Pi дозволяє програмно управляти потужним навантаженням з користувацької програми на ОС Raspberry. На платі встановлено три реле з максимальним струмом комутації до 10A і напругою до 250V [190]. Для індикації стану статусу реле в модулі передбачені світлодіоди. Загальний вид модуля представлений на рисунку 2.45.



Рисунок 2.45 – Загальний вид модуля 3-х реле для Raspberry Pi [190]

Особливості модуля реле для Raspberry Pi:

- сумісність із Raspberry Pi A+/B+/2B/3B/3B+;
- якісні реле з комутацією навантаження 250VAC/5A, 30VDC/5A;
- оптоізоляція для забезпечення якісної високовольтної ізоляції;
- вбудовані індикатори стану реле;

– перемички керування реле з можливістю підключення зовнішнього сигналу керування;

– підтримка засобами розробки в Wiringpi, Webiopi, Shell, Python і BCM2835.

Розташування основних елементів та їх призначення представлені на рисунку 2.46.

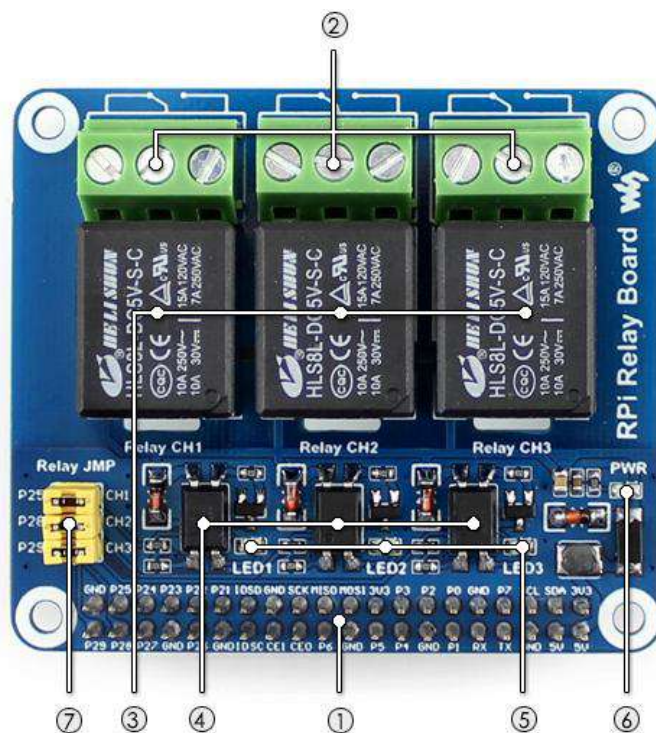


Рисунок 2.46 – Розташування основних елементів модуля 3-х реле для Raspberry Pi

Призначення елементів на модулі для 3-х реле для Raspberry Pi представлено на рисунку 2.46.

1. Raspberry Pi GPIO інтерфейс для підключення до Raspberry Pi;
2. Роз'єми контактів реле;
3. Реле;
4. Оптипара PC817;
5. Індикатор стану реле;
6. Індикатор світить: реле включене;

7. Індикатор не світить: реле виключене;
8. Індикатор живлення;
9. Перемичка керування реле;
10. Установлена: програмне керування реле;
11. Знята: керування реле перемичками або додатковими.

Модуль колекторних двигунів для Raspberry Pi на MC33886

Модуль драйвера двох колекторних моторів для Raspberry Pi Motor Driver Board на мікросхемах MC33886 [191]. Сполучно практично з усією лінійкою міні-комп'ютерів Raspberry Pi і прекрасно підходить для побудови автономних робо-платформ і роботів. На платі розміщений потужний стабілізатор напруги, що дозволяє живити Raspberry Pi від силових акумуляторів напругою до 40В. Загальний вид модуля на MC33886 представлено на рисунку 2.47.

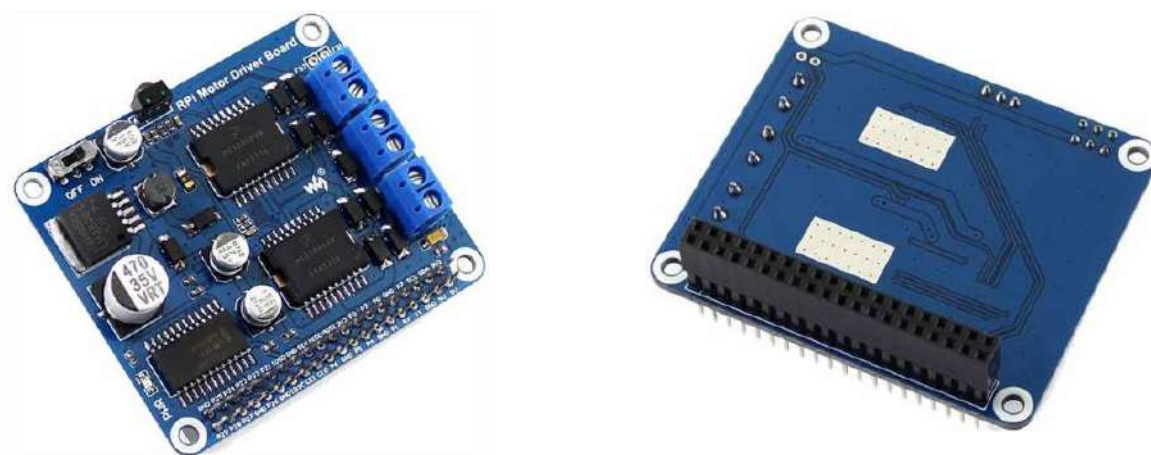


Рисунок 2.47 – Модуль драйвера двох колекторних моторів для Raspberry Pi Motor Driver Board на мікросхемах MC33886 [192]

Особливості модуля Motor Driver Board:

- сумісність із моделями Raspberry Pi A+/B+/2B/3B/3B+;
- мікросхема мостового драйвера двигуна: MC33886;
- вбудований стабілізатор напруги 5В для живлення Raspberry Pi;
- вбудований IR приймач для керування роботом;

- кілька захисних ланцюгів для стабільної роботи;
- запобіжник, що самовідновлюється;
- Driver chip features Short-Circuit Shutdown for Large Output Current, Undervoltage Disable Function;
- Reversed polarity protection for motor output;
- захист від неправильної полярності джерела живлення;
- діапазон вхідної напруги: 7В~40В;
- максимальне споживання одного мотора: до 5А;
- максимальний струм для живлення Raspberry Pi: до 2А.

Призначення основних елементів модуля Motor Driver Board [192] і їх опис представлено на рисунку 2.48.

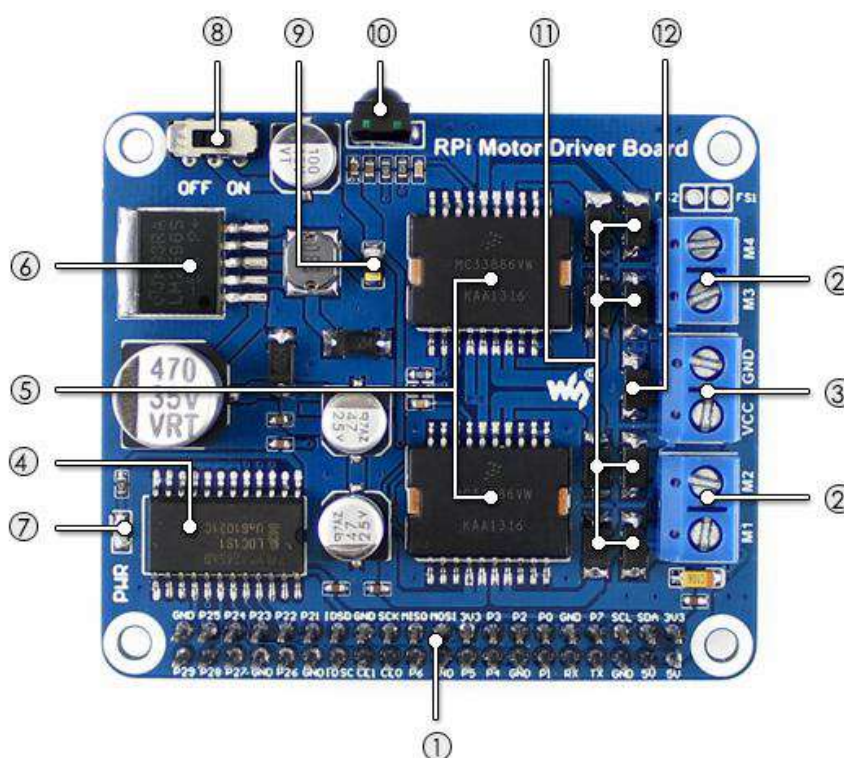


Рисунок 2.48 – Призначення загальних елементів модуля Motor Driver Board

1. Raspberry Pi GPIO інтерфейс для підключення до Raspberry Pi;
2. Роз'єми для підключення двигунів;
3. Вхід в зовнішнє джерело живлення;
4. Перетворювач рівня логічних сигналів 74LVC4245AD;
5. Мікросхема драйвера двигунів MC33886;

6. Мікросхема стабілізатора LM2596-5.0;
7. Індикатор живлення;
8. Перемикач живлення – Raspberry Pi:
 - OFF: Raspberry Pi живить драйвер двигуна;
 - ON: Драйвер двигателя живить Raspberry Pi;
9. Запобіжник, що самовідновлюється – 2А;
10. IR приймач;
11. Захисний діод Шоттки драйвера двигуна;

Модуль прецизійного АЦП/ЦАП для Raspberry Pi

При всіх перевагах міні-комп'ютерів у них є один істотний недолік, що стримує їхнє застосування – відсутність аналогових входів і виходів. І, якщо з останніми простіше – їх можна замінити ШІМ виходами, то із входами необхідне підключення додатково модуля АЦП/ЦАП Raspberry Pi High-Precision AD/DA Expansion Board. Він дозволяє підключити 8 вхідних аналогових сигналів, оцифровуючи їх з 24-розрядним дозволом і частотою до 30кГц, і 2 аналогових виходу з дозволом 16 розрядів і часом установлення напруги 10мкс. Аналогові входи можна використовувати як 4 диференціальних входів [193]. Джерело опорної напруги вбудоване, але є можливість підключення зовнішнього джерела. Загальний вид модуля прецизійного АЦП/ЦАП для Raspberry Pi представлено на рисунку 2.49.

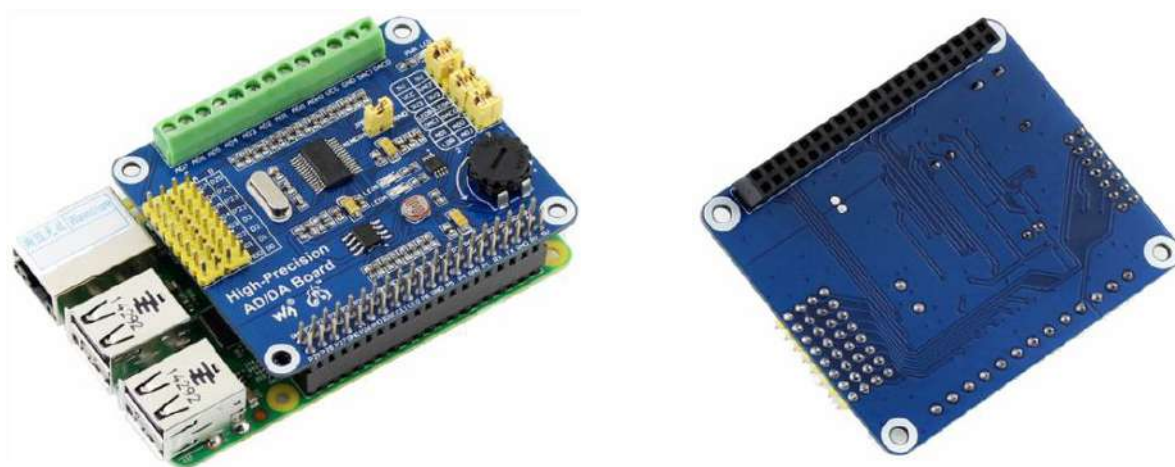


Рисунок 2.49 – Загальний вид модуля прецизійного АЦП/ЦАП [194]

Підтримувані моделі Raspberry Pi:

- Raspberry Pi 1 Model A+;
- Raspberry Pi 1 Model B+;
- Raspberry Pi 2 Model B;
- Raspberry Pi 3 Model B;
- Raspberry Pi 3 Model B+;

Особливості модуля прецизійного АЦП/ЦАП:[194]

- вбудований ADS1256 8-канальний 24-розрядний прецизійний АЦП (4 диференціальних входу), 30ksps частота оцифровки;
- вбудований DAC8532 2-канальний 16-розрядний прецизійний ЦАП;
- вбудований інтерфейс із роз'ємами для підключення аналогових сигналів;
- розміщення контактів сумісне з Waveshare стандартом датчиків;
- вбудований інтерфейс для підключення аналогових датчиків " під гвинт";
- вбудовані датчики й індикатори для демонстрації обладнання.

Розміщення основних елементів і їх опис представлено на рисунку 2.50.

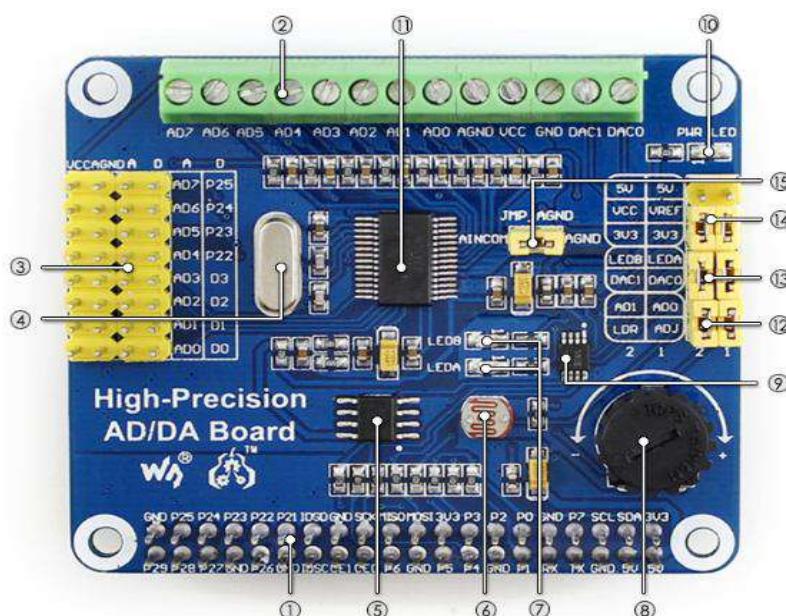


Рисунок 2.50 – Розміщення основних елементів на модулі прецизійного АЦП/ЦАП [194]

1. Роз'єм розширення RPi для підключення до міні-комп'ютера;
2. АЦП/ЦАП входи/виходи;
3. АЦП входи. Розміщення контактів сумісне з Waveshare стандартом датчиків;
4. Кварцовий резонатор 7.68МГц;
5. Мікросхема джерела опорної напруги LM285-2.5 для АЦП;
6. Фоторезистор;
7. Світлодіоди стану виходів ЦАП;
8. Змінний резистор 10КОм;
9. Мікросхема прецизійного ЦАП DAC8532;
10. Світлодіодний індикатор живлення;
11. Мікросхема прецизійного АЦП ADS1256;
12. Тестові перемички АЦП;
14. Тестові перемички ЦАП;
15. Перемички вибору джерела живлення;
16. Перемикач конфігурації ADC: використання вбудованого джерела напруги або зовнішнього, що підключається до контакту AINCOM.

Комунікаційний модуль GSM/GPRS/GNSS/Bluetooth HAT для Raspberry Pi

Комунікаційний модуль для Raspberry Pi GSM/GPRS/GNSS/Bluetooth HAT на модулі SIM868 з підтримкою GSM, GPRS, GNSS і Bluetooth, що дозволяє легко перетворити Raspberry у мобільний телефон, меседжер, GPS-трекер, сервер збору й передачі інформації з датчиків і багато чого іншого. Даний шилд сумісний з Raspberry Pi 3B+ / 3B / 2B / Zero/Zero W. За допомогою комунікаційного шилду можна управляти об'єктом на будь-якій відстані, одержувати й передавати дані [195]. Управляється й налагоджується шилд за допомогою вбудованого USB-UART CP2102 (рис. 2.51).

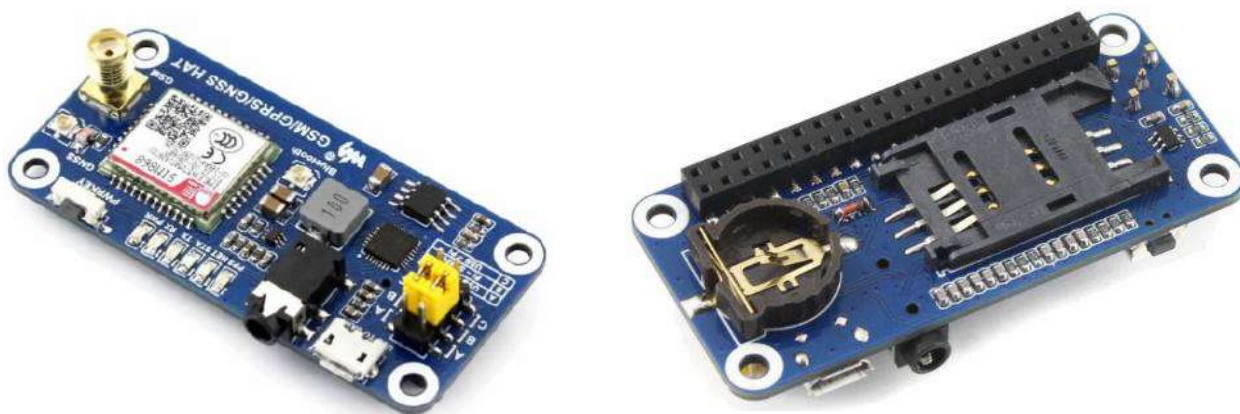


Рисунок 2.51 – Комунікаційний модуль GSM/GPRS/GNSS/Bluetooth HAT для Raspberry Pi [195]

Основні можливості комунікаційного модуля GSM/GPRS/GNSS/Bluetooth HAT для Raspberry Pi:

- повна сумісність із інтерфейсом Raspberry Pi 2B/3B/3B+/Zero/Zero W;
- підтримка приймання й відправлення SMS, телефонних дзвінків, приймання й передачі даних, GPRS, DTMF, HTTP, FTP, MMS, email і т.д.;
- підтримка GPS, COMPASS, Glonass, LBS позиціонування по базових станціях;
- підтримка передачі даних через Bluetooth 3.0;
- USB-UART конвертер на CP2102 для налагодження програм;
- 6 світлодіодів режиму роботи модуля;
- слот для SIM карти 1.8V/3V;
- наявність батарежки годин реального часу;
- автовизначення швидкості послідовного порту (від 1200 бод до 115200 бод);
- керування AT командами (3GPP TS 27.007, 27.005 і розширені SIMCOM AT команди);
- підтримка додатків SIM application toolkit: GSM 11.14 Версія 99;
- підтримка ресурсами для розробки із прикладами програм Raspberry Pi/Arduino/STM32.

Призначення основних елементів комунікаційного модуля GSM/GPRS/GNSS/Bluetooth HAT для Raspberry Pi (рис. 2.52) описані нижче:

1. Модуль SIM868;
2. Стабілітрон ZMM5V1;
3. Діод SMF05C;
4. USB-UART конвертер CP2102;
5. Перетворювач напруги MP1482;
6. Перетворювач рівнів NDC7002N;
7. Індикатор стану GPS;
8. Індикатор стану мобільної GSM мережі:
 - швидке миготіння – початок роботи модуля;
 - повільне миготіння – успішна реєстрація в мобільній GSM мережі;
9. Індикатор стану STA модуля;
10. Індикатор стану SIM868 UART Tx/Rx;
11. Індикатор живлення;
12. SIM868 control button: press the button and hold for 1s, to startup/shutdown the SIM868;
13. Роз'єм підключення Raspberry Pi GPIO;
14. Роз'єм Sim-Карті;
15. USB-UART інтерфейс;
16. 3.5мм роз'єм підключення навушників і мікрофона;
17. Роз'єм підключення GNSS антени;
18. Роз'єм підключення Bluetooth антени;
19. Роз'єм підключення GSM антени;
20. CR1220 battery holder: for RTC backup battery;
21. Перемикач режиму роботи UART інтерфейсу.

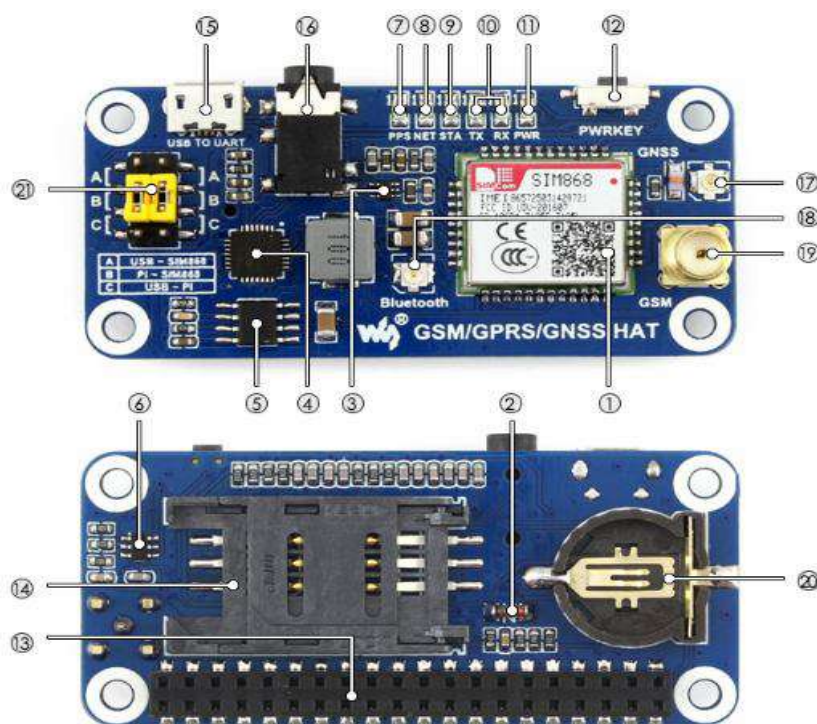


Рисунок 2.52 – Основні елементи комунікаційного модуля GSM/GPRS/GNSS/Bluetooth HAT для Raspberry Pi

Модуль годин реального часу для Raspberry Pi та Arduino

Модуль має відмінну точність ходу, тому що кварцовий резонатор розміщений безпосередньо в корпусі мікросхеми та застосовується цифрова корекція точності ходу годин залежно від температури навколишнього середовища. Модуль можна використовувати й з контролерами Arduino, підключивши його перемичками до контролера [196]. Загальний вид модуля представлено на рисунку 2.53.



Рисунок 2.54 – Модуль годин реального часу для Raspberry Pi [196]

Основні характеристики модуля годин реального часу для Raspberry Pi [197]:

- мікросхема годин: DS3231;
- напруга живлення: від 2.3 В до 5.5 В;
- робоча температура: від -40°C до $+85^{\circ}\text{C}$;
- точність ходу годин: $\pm 5\text{ppm}$ (± 0.432 сек/день) ;
- резервне живлення: Lipo акумулятор;
- параметри часу: годинник і календар (включаючи секунди, хвилини, годинник, дні, дату, місяць і рік до 2100 з урахуванням високосного року);
- сигнали календаря: два сигнали;
- цифрові виходи: 1 Гц і 32.768 кГц;
- інтерфейс: I2C 400 кГц;
- адреса обладнання: 0x68;
- точність датчика температури: $\pm 3^{\circ}\text{C}$.

1.9 Датчики

3-осьовий акселерометр GY-291 ADXL345

Акселерометр ADXL345 – це крихітний мікропотужний трьохосьовий акселерометр високого дозволу (13 біт). Діапазоном виміру положення коливається до $\pm 16\text{ g}$. Результат виміру віддається у вигляді 16-розрядних чисел у додатковому коді й через цифрові інтерфейси SPI/I²C [198]. Загальний вид 3-осьовий акселерометр GY-291 ADXL345 представлено на рисунку 2.55.

Даний акселерометр ADXL345 ідеально підходить для використання в мобільних обладнаннях – він вимірює статичне прискорення (викликане гравітацією) у завданнях визначення відхилення, або динамічне прискорення, викликане рухом або ударами. Заявлений високий дозвіл акселерометра ($4 \cdot 10^{-3}\text{ g/LSB}$) дозволяє точно відслідковувати зміни відхилення менш ніж

на 1.0°. Режим зниженого енергоспоживання датчика дозволяє реалізовувати інтелектуальне управління живленням системи [199].

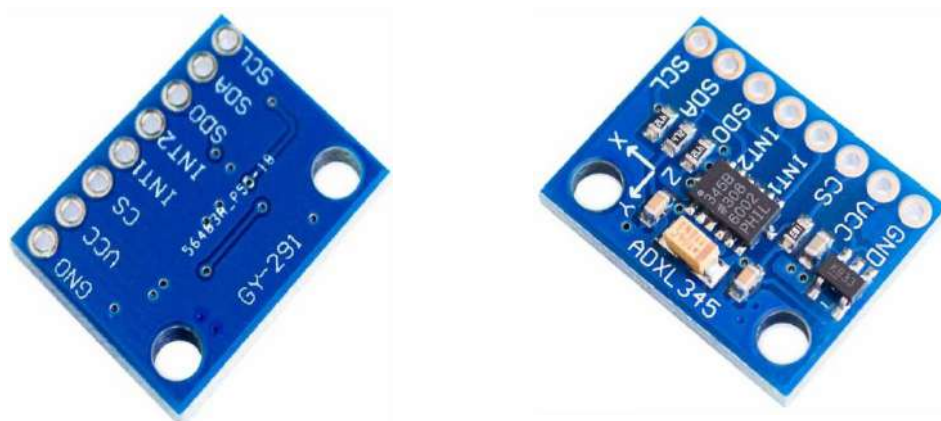


Рисунок 2.55 – Загальний вид 3-осьовий акселерометр GY-291 ADXL345 [198]

Основні характеристики 3-осьового акселерометра GY-291 ADXL345:

- живлення: 3-5В;
- підтримувані інтерфейси: I²C, SPI;
- дозвіл: 13bit, 4 мг / LSB;
- діапазон вимірів: + / - 16g;
- розміри: 20x15мм.

Особливості одноплатних комп'ютерів сімейства Raspberry Pi полягає в тому, що його можливостей вистачить для роботи з 90% датчиків. У цьому розділі ми розглянемо тільки ті датчики які будуть необхідні для розробки мобільних робототехнічних платформ.

IR датчик відстані (10-80 см) Sharp GP2Y0A21YK0F.

Сенсор із цифровою обробкою інформації й аналоговим виходом, що вимірює відстань у діапазоні від 10 до 80 см. Відрізняється високою точністю й мінімальним часом відгуку [200]. Загальний вид IR датчик представлено на рисунку 2.56.



Рисунок 2.56 – IR датчик відстані Sharp Sharp GP2Y0A21YK0F [200]

Основні характеристики IR датчик відстані Sharp GP2Y0A21YK0F [201]:

- робоча напруга: 4,5В – 5,5В;
- середній струм споживання: 30 мА;
- вимірювана відстань: 10 – 80 див;
- тип вихідного сигналу: аналоговий;
- середній час відновлення інформації на вивід OUT: 16,5 мс;
- розміри: 29,5 x 13 x 13,5 мм [90].

IR датчик відстані (4-30 см) Sharp GP2Y0A41SK0F

Маленький сенсор із цифровою обробкою інформації й аналоговим виходом, що вимірює відстань у діапазоні від 4 до 30 см. Відрізняється високою точністю, мінімальним часом відгуку, компактним розміром і низьким споживаним струмом [202]. Загальний вид представлено на рисунку 2.57.

Основні характеристики IR датчик відстані GP2Y0A41SK0F [203]:

- робоча напруга: 4,5В - 5,5В;
- середній струм споживання: 12 мА;
- вимірювана відстань: 4 - 30 см;
- тип вихідного сигналу: аналоговий;
- середній час відновлення інформації на вивід OUT: 16,5 мс;
- розміри: 29,5 x 13 x 13,5 мм [91].

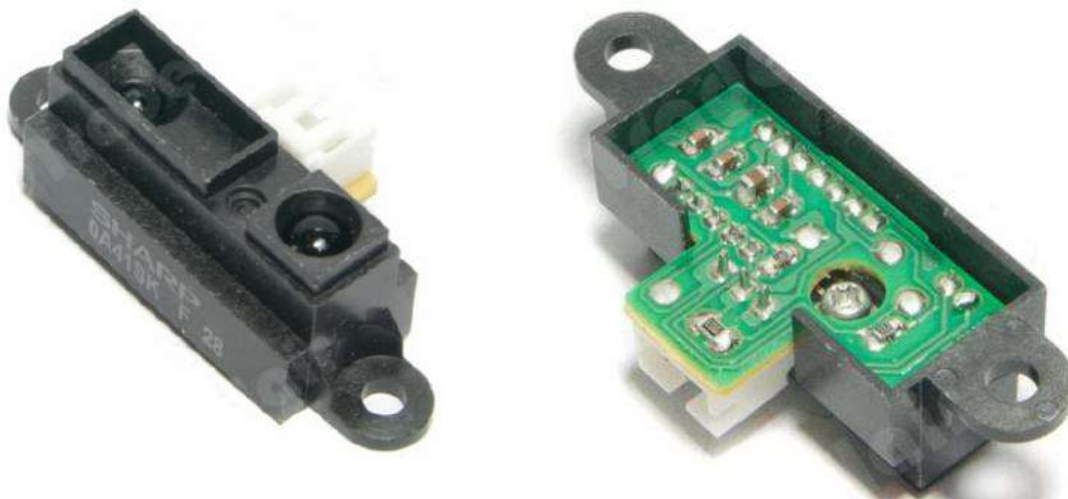


Рисунок 2.57 – IR датчик відстані (4-30 см) Sharp GP2Y0A41SK0F [202]

Аналоговий датчик ультрафіолету на GUVVA-S12SD

Модуль датчика ультрафіолету на фотодіоді GUVVA-S12SD з піком чутливості в діапазоні випромінювання 240-370нм (UVB і більша частина UVA) ідеально підходить для конструкцій, що вимірюють рівень ультрафіолетового випромінювання. Робочий струм фотодіода дуже малий – наноампери, тому для підключення його до аналогового входу контролерів потрібне застосування підсилювача на операційній системі [204]. Загальний вид представлено на рисунку 2.58.

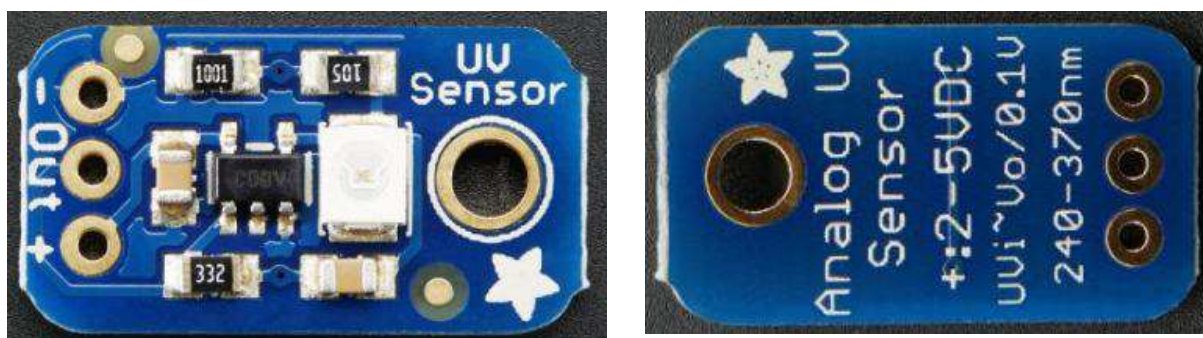


Рисунок 2.58 – Загальний вид аналогового датчика ультрафіолету на GUVVA-S12SD [205]

Вихідна напруга датчика розраховується за формулою: $V_{out} = 4.3 \cdot I$ (де I – струм діода). Так при струмі у 1мкА ($9\text{мВт}/\text{кВ}$) вихідна напруга складе 4.3 В. При перерахуванні напруги в індекс УФ випромінювання потрібно розділити вихідну напругу на 0,1В, тобто якщо вихідна напруга складе 0,5 В, індексі УФ випромінювання дорівнюватиме 5. Зверніть увагу на те, що УФ світлодіоди, як правило, мають інший діапазон випромінювання. Для реєстрації випромінювання й перевірки датчика краще підійдуть сонячне випромінювання та лампи для засмаги.

Основні характеристики аналогового датчика ультрафіолету на GUVA-S12SD [204, 205]:

- тип фотодіода: GUVA-S12SD;
- тип виходу: аналоговий;
- робочий діапазон УФ випромінювання: 240-370нм (UVB і більша частина UVA);
- напруга живлення: від 2.7 В до 5.5 В;
- розміри: 10 мм х 19 мм х 2 мм.

Водонепроникний ультразвуковий далекомір DYP-ME007 (30 см-3.5 м)

DYP-ME007 – водонепроникний ультразвуковий модульний далекомір, що забезпечує вимір відстані в діапазоні від 30 см до 3.5 м. Далекімір застосовується для виміру таких відстаней як: автомобільний вимірник відстані при паркуванні, відхилення від перешкод роботів, системи безпеки й промислового контролю та багатьох інших [206]. Загальний вид ультразвукового далекоміра DYP-ME007 представлено на рисунку 2.59.

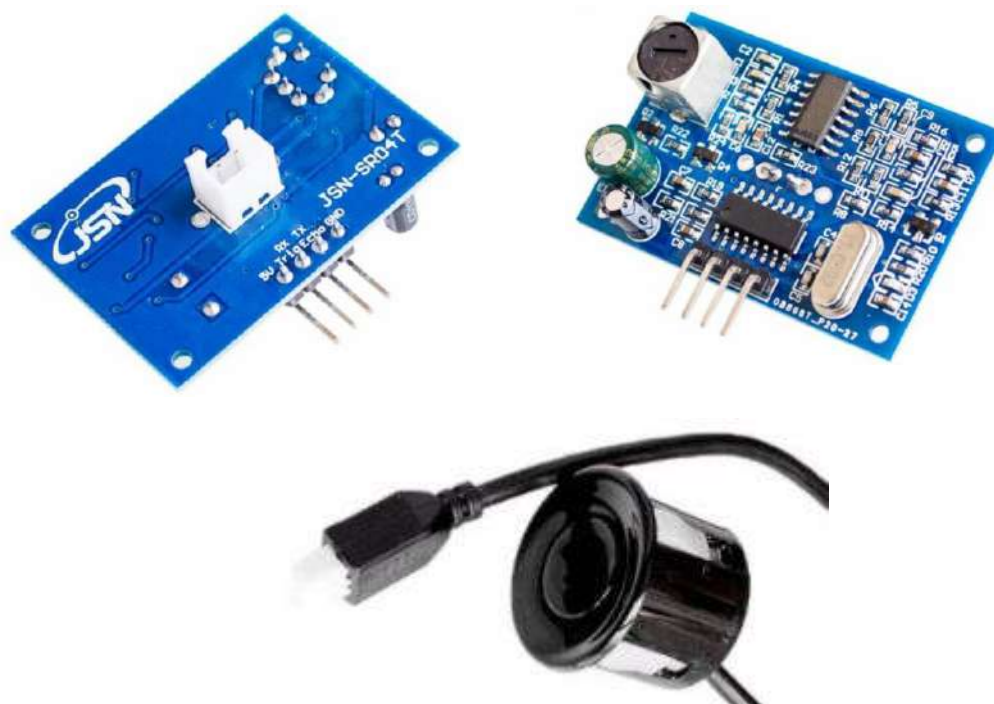


Рисунок 2.59 – Загальний вид ультразвукового далекоміра DYP-ME007 [207]

Основні технічні характеристики ультразвукового далекоміра DYP-ME007 [207]:

- робоча напруга: DC 5 В;
- робочий струм: <15 ма;
- частота: 40 кГц
- діапазон виміру: 30 см - 3.5 м;
- вихідний сигнал: 5 В рівень TTL;
- час відгуку: 100 мс;
- точність: 1 см;
- робоча температура: -10 - 60 C⁰;
- температура зберігання: -20 - 80 C⁰;
- розмір (Д x Ш): ≈ 4x3 см;
- довжина кабелю: ≈ 200 см;
- діаметр випромінювача 12 мм, корпуса 25 мм, висота корпуса 20 мм.

Датчик вологості й температури SHT10

Цифровий датчик температури й вологості SHT1x простий у використанні, точний і підключається стандартним інтерфейсом I²C. Якісне калібрування забезпечує високу точність, відмінну стабільність і довгочасне використання [208]. Загальний вид датчик вологості й температури SHT10 представлено на рисунку 2.60.

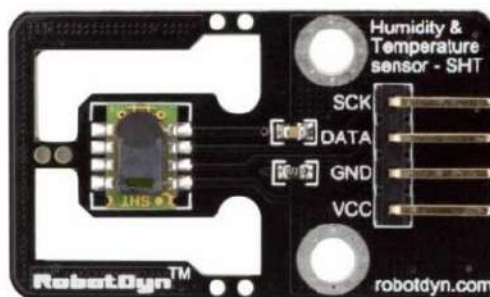


Рисунок 2.60 – Загальний вид датчик вологості й температури SHT10

Технічні характеристики датчика вологості й температури SHT10 [208]:

- напруга живлення: 3,5 – 5 В;
- низьке енергоспоживання 30 м Вт;
- вимірювана вологість: 0 ... 100% відносної вологості;
- абсолютна точність виміру вологості +/- 2% відносної вологості (відносна вологість 10-90%);
- визначається температурою: -40 - + 128°S ± 0,3%;
- точність: +/- 0,3 ° С при 25 ° С;
- частота дискретизації: ≤ 1 Гц [94].

Датчик руху мікрхвильовий RCWL-0516

Датчик руху RCWL-0516 працюючий на ефекті Доплера, з можливістю виявлення рухів минаючи перешкоди, використовується з Ардуїно або іншими платформами. Також його можна застосовувати прямо з релейними

модулями [209]. Загальний вид датчика руху RCWL-0516 представлено на рисунку 2.61.



Рисунок 2.61 – Загальний вид датчика руху RCWL-0516

В основу роботи датчика закладений ефект Доплера – зміна частоти відбитої хвилі, внаслідок руху випромінювача, приймача або відбивача. У даному модулі частота випромінюваної ним радіохвилі змінюється внаслідок руху відбивача (перешкоди). Модуль побудований на базі чипу RCWL-9196 оснащеного передавачем і приймачем. Датчик спрацює, якщо приймач прийме сигнал, частота якого незначно відрізняється від частоти сигналу передавача:

- якщо в зоні дії датчика немає об'єктів здатних відбивати радіохвилі, то приймач нічого не прийме й датчик не спрацює;
- якщо в зоні дії датчика є нерухливі об'єкти здатні відбивати радіохвилі, то приймач прийме радіохвилю передавача, відбиту від цих об'єктів, але частота прийнятої радіохвилі буде дорівнювати частоті сигналу передавача й датчик не спрацює;
- якщо в зоні дії датчика є об'єкт здатний відбивати радіохвилі, який наближається до датчика (рухається), то приймач прийме відбиту від об'єкта радіохвилю, частота якої буде вище чому в сигналу передавача й датчик спрацює;
- якщо в зоні дії датчика є об'єкт здатний відбивати радіохвилі, який віддаляється від датчика (рухається), то приймач прийме відбиту від об'єкта радіохвилю, частота якої буде нижче ніж в сигналі у передавача й датчик спрацює.

Коли датчик спрацьовує, на його виході «OUT» установлюється рівень логічної «1». Датчик оснащений тригером, який утримує рівень логічної «1» на виході «OUT» протягом 2 сек $\pm 30\%$, після припинення руху.

Якщо датчик багаторазово спрацьовує, наприклад, постійно фіксує рух протягом 10 секунд, то рівень логічної «1» на виході «OUT» буде встановлений на 12 секунд із моменту першого спрацьовування (10 секунд під час фіксації рухів + 2 секунди після їхнього припинення, поки не «скинеться» тригер). Використання датчиків руху заснованих на ефекті Доплера дозволяє фіксувати рух через об'єкти, що не відбивають радіохвилі (дерево, пластик, гіпс і т.д.), чого не можуть зробити датчики руху засновані на піроелектричному ефекті (такі як HC-SR501) [95].

Підключення датчика руху RCWL-0516 [211]:

OUT – вихід датчика (установлюється в «1» при наявності рухів + затримується на 2 секунди після їх припинення).

VIN – вхід напруги живлення, від +4 до +28 В постійного струму.

GND – вхід живлення (загальний).

3V3 – вихід стабілізованої напруги живлення 3,3 В (можна використовувати для живлення мікроконтролерів).

COS – вхід дозволу (підтягнутий внутрішнім опором чіпа). Якщо на даному виводі встановити рівень логічного «0», то після скидання тригера, він не буде встановлюватися (датчик перестане реагувати на рухи).

Вихід «OUT» підключається до будь-якого виводу Arduino (на схемі підключений до виводу D4). Вхід «COS» можна залишити не підключеним, тому що він підтягнутий внутрішнім опором чипу, отже, дозволяє роботу тригера. Додатковий вивід CDS можна використовувати для дистанційної установки фоторезистора, а так само для добору змінним резистором ступеню освітленості, при якій починає працювати датчик, потім цей номінал припадає до R-CDS.

Основні параметри датчика руху RCWL-0516:

– напруга живлення (VIN): 4-28 В постійного струму;

- споживаний струм: до 3 мА (номінально 2,8 мА);
- дальність виявлення: до 9 м (номінально до 5 м);
- кут виявлення: 120 град;
- потужність передавача: до 30 мВт (номінально до 20 мВт);
- частота передавача: 5,8 ГГц;
- час затримки до скидання тригера: 2 сек $\pm 30\%$;
- вихідна напруга (3V3): 3,2...3,4 В (номінально 3,3 В);
- максимальний струм на виході «3V3»: до 100 мА;
- робоча температура: -20 до +80 °С;
- температура зберігання: -40 до +100 °С.

Датчик лінії для Arduino від Robotdyn.

Сенсор відстеження лінії для Arduino може виявляти білі лінії на чорному фоні й чорні лінії на білому фоні. Якщо датчик відстеження лінії виявляє чорний колір, сигнальна лінія (DO) датчика переходить у стан HIGH (1), у той же час на датчику загоряється світлодіод (синій) [211]. Якщо датчик відстеження лінії виявляє білий колір, сигнальна лінія (DO) переходить у стан LOW (0). За допомогою потенціометра можна налаштувати чутливість датчика лінійного спостереження й відстань до лінії. Загальний вид датчик лінії представлено на рисунку 2.62.



Рисунок 2.62 – Датчик лінії для Arduino від Robotdyn [212]

Основні характеристики датчика лінії:

- напруга живлення: 3.3 В / 5 В;
- споживаний струм: 20 мА при 5 В;
- вага: 2,8 г [96].

Датчик освітленості GY-49 MAX44009 I²C 3.3V

Датчик освітленості на мікросхемі MAX44009 (GY-49) призначений для виміру рівня навколишнього висвітлення наближеного до чутливості людського ока. Модуль має цифровий вихід через інтерфейс I²C [213]. Датчик за характеристиками відмінно підходить для портативних пристроїв або промислового призначення. Наявність на платі стабілізатора напруги живлення й перетворювача рівнів інтерфейсу I²C дозволяє використовувати модуль із мікроконтролерами як з 5 В так і з 3,3 В логікою. Загальний вид датчика GY-49 MAX44009 представлений на рисунку 2.63.

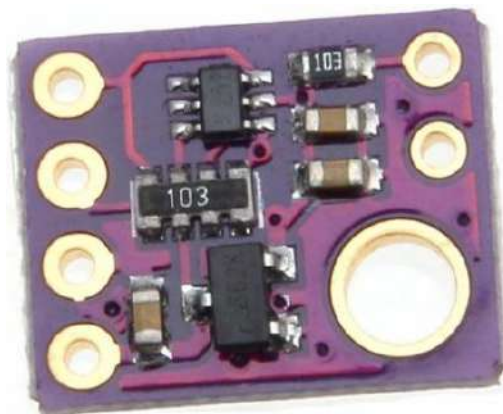


Рисунок 2.63 – Загальний вид датчика
GY-49 MAX44009 [214]

Спектральний відгук чипа оптимізований робити коректні виміри в режимі природного висвітлення й містить у собі ІЧ-фільтр і УФ-фільтр для усунення засвічення при штучному освітленні. Блок адаптивного посилення автоматично обирає коректний діапазон освітленості для оптимізації значень. Програмувальний вихід переривання виконаний у вигляді виходу з відкритим стоком і може обслуговувати керуючу логіку з будь-яким діапазоном напруг живлення.

Основні характеристики датчика GY-49 MAX44009 [214]:

- інтерфейс: I²C;
- діапазон освітленості: 0.045 - 188000 Лк;

- динамічний діапазон: 22 біт;
- живлення: 3,3 - 5 В;
- струм споживання: 0.65 мкА;
- розмір плати: 2 х 1 см;
- робоча температура: - 40 до +85 °С.

Датчик пульсу MAX30102

Датчик серцевого ритму (пульсу) MAX30102 – інтегральний датчик пульсу й насичення крові киснем. У датчику зібрано оптимізовану оптику, два світлодіоди, фотодетектор, високоточний аналоговий підсилювач і перетворювач, цифровий оброблювач і інтерфейсний модуль. Для підключення датчика до контролера використовується послідовний інтерфейс I²C [215]. Загальний вид датчика MAX30102 представлено на рисунку 2.64.

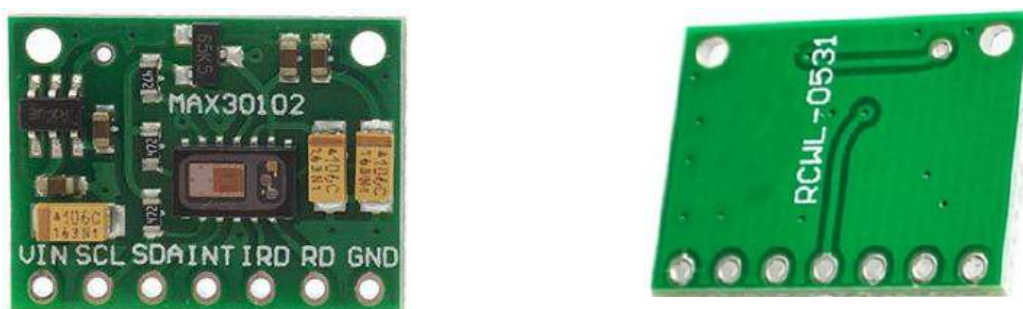


Рисунок 2.64 – Загальний вид датчика MAX30102 [216]

У датчику використовується два світлодіоди: червоного й інфрачервоного спектра. Є можливість керувати як струмом через світлодіоди (від 0 мА до 50 мА) так і тривалістю імпульсів (від 200 мкс до 1,6 мс) для забезпечення максимальної точності вимірів. Для калібрування датчика є вбудований датчик температури.

Характеристики датчика MAX30102 [216]:

- вимірювані параметри: частота пульсу й насичення крові й SpO₂;
- напруга живлення: 3,3 В (внутрішній стабілізатор на 1,8 В);

- споживаний струм у режимі виміру: 1.2 мА;
- споживаний струм у режимі сну: до 10 мкА;
- інтерфейс: I²C;
- максимальна частота інтерфейсу: 400 кГц.

Датчик температури й вологості SHT31-D ІІС.

Модуль цифрового датчика температури й вологості SHT-31D відкриває новий рівень у технології вимірів. У функціонал датчика входить схема обробки й посилення сигналу, блок пам'яті калібрування, АЦП і схема скидання по живленню, а також широкий діапазон напруги живлення від 2.5 до 5.5В. Даний модуль може використовуватися з різними мікроконтролерами на базі Arduino або Raspberry Pi [217]. Загальний вид представлено на рисунку 2.65.



Рисунок 2.65 – Датчик температури й вологості SHT31-D ІІС [218]

Основні характеристики датчика температури й вологості SHT31-D ІІС [218]:

- діапазон виміру відносної вологості: 0 - 100 %;
- точність діапазону відносної вологості: 2 %;
- тип інтерфейсу: I²C;
- дозвіл: 16 біт;
- точність: +/- 0.3 °C;
- напруга живлення: 2.5 - 5.5 В;

- робочий струм: 800 мА;
- робоча температура: -40 до +125 °С.

Датчик струму з аналоговим виходом на MAX471.

Модуль аналогового типу для виміру сили струму споживаного навантаженням від джерела живлення. Принцип виміру заснований на контролі напруги, що знімається із прецизійного дільника напруги. Для виміру використовується мікросхема MAX471 із вбудованим вимірювальним шунтом [219]. Загальний вид представлено на рисунку 2.66.

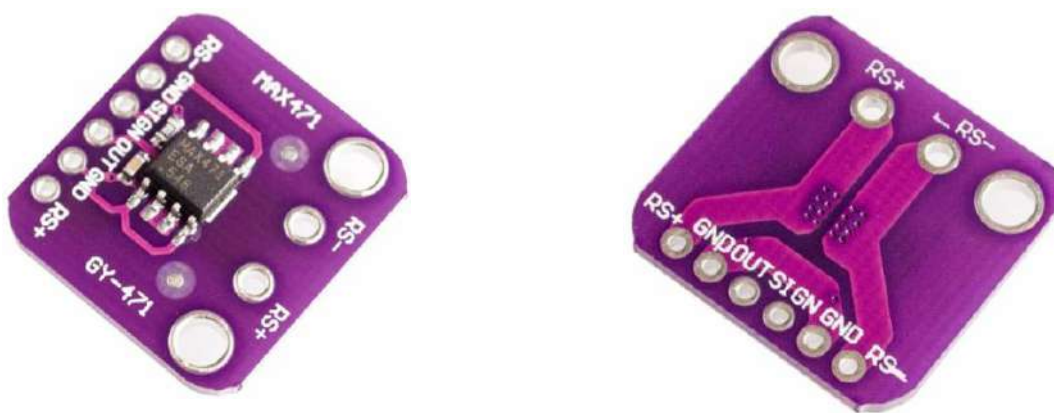


Рисунок 2.66 – Загальний вид датчик струму з аналоговим виходом на MAX471 [219]

Максимальний вимірюваний струм для одного модуля становить 3 А, для збільшення цієї межі, схема дозволяє застосовувати паралельне включення. Або можна використовувати зовнішній струмовий резистор.

Основні характеристики датчика струму з аналоговим виходом на MAX471 [220]:

- напруга живлення MAX471: від 3 до 36 В;
- вимірюваний струм: від 0 до 3 А;
- точність виміру струму: 2%;
- напрямок виміру струму: двонаправлений;
- споживаний струм: 100 мкА;
- споживаний струм у режимі очікування: 18 мкА;

- корпус MAX471: 8-вивідний SOIC;
- розмір друкованої плати: 22 x 21 x 7 мм [100].

Датчик ультрафіолету на GUVVA-S12SD

Модуль датчика ультрафіолету на GUVVA-S12SD ідеально підходить для конструкцій, що вимірюють рівень ультрафіолетового випромінювання. Лінійна залежність датчика вимірів дозволяє обійтися без перетворень, що знижують швидкість і точність вимірів [221]. Загальний вид датчика ультрафіолету на GUVVA-S12SD представлено на рисунку 2.67.

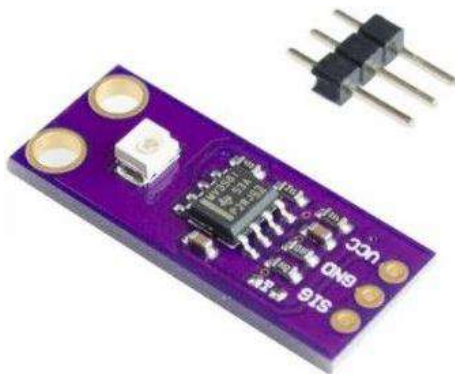


Рисунок 2.67 – Загальний вид датчика ультрафіолету на GUVVA-S12SD

Основні характеристики датчика ультрафіолету на GUVVA-S12SD [222]:

- датчик: GUVVA-S12SD;
- діапазон вимірів: 240 нм-370 нм;
- великий кут реєстрації: 130 градусів;
- тип світлочутливих елементів: діоди Шотткі;
- напруга живлення: 2.5 В ~ 5 В;
- робочий струм становить: 1мА;
- тип характеристики: лінійна.

ІК-термомодуль (4-pin)

ІК-термомодуль призначений для виміру температури повітря й/або предметів. Маленькі розміри, висока точність і можливість прямо підключати його до контролера Arduino або модулю реле. Дозволяють широко використовувати його в конструкціях де потрібно точно й швидко вимірювати температуру: термометри, термостати, системи вентиляції й кондиціонування повітря [223]. Загальний вид представлено на рисунку 2.68.



Рисунок 2.68 – Загальний вид ІК-термомодуля (4-pin) [223]

Основні характеристики й особливості модуля [224]:

- використовується NTC чуттєвий термістор;
- діапазон температури, що вимірюється від 20 до 80 °C;
- використовується NTC чутливий термістор;
- цифровий вихід зі струмом до 15 мА;
- аналоговий вихід сигналу датчика;
- поріг, що настраюється, спрацьовування цифрового виходу;
- вбудований індикатор стану цифрового виходу й напруги живлення;
- робоча напруга від 3 до 5 В;
- розмір: 3.2 см x 1.4 см;
- використовується розповсюджений компаратор LM393 із широким діапазоном робочої напруги.

Лазерний датчик відстані GY-530 на VL53L0X

GY-530 VL53L0X – мініатюрний модуль датчика відстані й розпізнавання жестів. Швидкий і точний вимір відстані до 2 м. Модуль підключається через розповсюджений послідовний інтерфейс I²C для

керування обладнанням і передачі даних. Напруга живлення модуля 5 В [225].

VL53L0X оснащений матрицею на основі високочутливих однофотонних лавинних діодів. Принцип роботи заснований на запатентованій Stmicroelectronics технології Flightsensetm. Загальний вид представлено на рисунку 2.69.



Рисунок 2.69 – Лазерний датчик відстані GY-530 [226]

Поверхнево-випромінюючий лазер VCSEL з довжиною хвилі 940 нм, виконує в датчику відстані VL53L0X роль джерела оптичного сигналу. Він оснащений вбудованим інфрачервоним фільтром. Його світіння повністю невидимо для людського ока і забезпечує більшу дистанцію виміру при меншій чутливості, до рівня зовнішнього освітлення і є більш стійкий до перехресних перешкод, що викликано скляними поверхнями. Датчик VL53L0X має вкрай низьке енергоспоживання.

Можливі області застосування лазерного датчика відстані GY-530 [226]:

- датчики присутності персональних ПК / ноутбуків / планшетних ПК і обладнань Інтернету речей (автоматичні вмикачі освітлення);
- робототехніка (датчики виявлення перешкод);
- побутова техніка (автоматичні крани, вентиля, дозатори мила і т.д.);
- одномірні датчики розпізнавання жестів;

– лазерні системи автофокусування. Поліпшує й прискорює роботу автофокусу камери, особливо в несприятливих умовах (низька освітленість, низька контрастність) або при зйомці на високій швидкості руху [102].

Модуль датчика RGB і жестів APDS-9960

Датчик жестів RGB APDS-9960 – це цифровий датчик RGB навколишнього освітлення, датчик наближення й жестів. Датчик оснащений інтерфейсом, сумісним з I²C, із червоним, зеленим, синім, прозорим (RGBC) сенсорами наближення й жестів з IR-світлодіодом. Функція виміру RGB і навколишнього освітлення визначає інтенсивність світла при різних умовах освітлення й різних матеріалах, включаючи затемнене скло. Крім того, вбудований УФ-IR-Фільтр блокує паразитну IR і УФ засвічення й корелює передачу кольору [227]. Загальний вид модуля датчика RGB і жестів APDS-9960 представлено на рисунку 2.70.

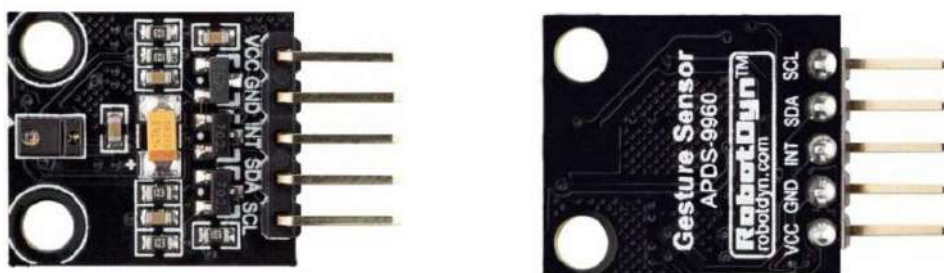


Рисунок 2.70 – Загальний вид модуля датчика RGB і жестів APDS-9960 [228]

Функція наближення й жестів відкалібрована до 100 мм відстані виявлення наближення, не вимагаючи додаткового калібрування датчика. Виявлення жесту використовує чотири спрямовані фотодіоди, інтегровані з видимим фільтром, для точного визначення простих жестів униз-вправо-уліво або більш складних жестів. Додавання мікрооптичних лінз у модуль забезпечує високу ефективність передачі й приймання ІЧ-випромінювання. Внутрішній контролер переводить датчик у режим зниженого

енергоспоживання між вимірами RGBC наближення й жестів, забезпечуючи дуже низьке енергоспоживання.

Особливості модуля датчика RGB і жестів APDS-9960 [228]:

- датчик світла RGBC, детектор наближення й жестів з IR-Індикатором в оптичному модулі;
- інтерфейс сумісний з I²C з виводом спеціального переривання;
- висока чутливість, що забезпечує роботу із темним склом;
- світлочутливість RGBC із вбудованим блоковим фільтром UV-IR;
- геометрично впорядковані фотодіоди RGBC, що забезпечують рівномірний кутовий відгук;
- калібрування до 100-міліметрової відстані виявлення;
- чотири окремі фотодіоди, чутливі до різних напрямків;
- вимір наближення й жест із інтегрованим блоковим фільтром;
- запатентована конструкція екрану;
- низьке енергоспоживання: 1,0 мкА, типовий у режимі сну [103].

Модуль датчика звуку

Модуль датчика звуку від WaveShare побудований на звуковому підсилювачі мікрофона LM386 і компараторі LM393. Служить для одержання інформації про звуковий фон навколишнього середовища в мікроконтролер, або використання для підключення до звукового підсилювача через цифровий або аналоговий виходи, з метою оцінки та виміру його інтенсивності. Легко підключається до контролера перемичками, які входять у комплект модуля. Коефіцієнт підсилення й поріг спрацьовування регулюються змінними резисторами на платі [229]. Загальний вид представлено на рисунку 2.71.

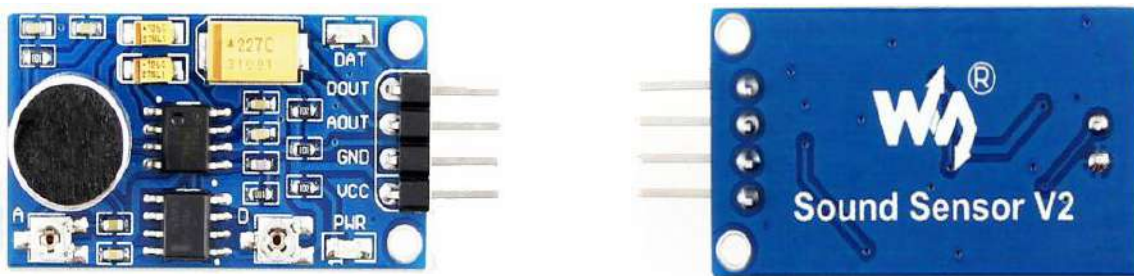


Рисунок 2.71 – Загальний вид модуля датчика звуку від Waveshare [229]

Загальні характеристики модуля датчика звуку [230]:

- чутливість мікрофона 52 дБ;
- частотний діапазон 50 Гц ~ 20 КГц;
- операційний підсилювач LM386;
- коефіцієнт підсилення 200;
- напруга живлення 3.3 ~ 5.3 В;
- діаметр монтажних отворів: 2.0 мм [104].

Модуль датчика наближення VL6180X

Модуль унікального датчика наближення й освітленості VL6180X, що використовує технологію Flightsense від Stmicroelectronics, здатний визначати відстань до об'єктів у діапазоні відстані 1-100 мм із міліметровою точністю, незалежно від характеристик об'єкта, а також вимірювати освітленість у широкому динамічному діапазоні. Переваги нового датчика дозволяють зробити прорив в області побудови безконтактних систем керування як у побутовій і портативній електроніці, так і в промислових системах [231]. Загальний вид представлено на рисунку 2.72.

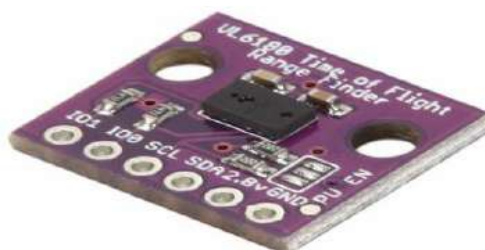


Рисунок 2.72 – Загальний вид модуль датчика наближення VL6180X [232]

Загальні характеристики датчика наближення VL6180X [232]:

- тип випромінювача – інфрачервоний;
- довжина хвилі випромінювача – 850нм;
- вимірювана відстань – 0-50 см;
- тип інтерфейсу – I²C;
- адреса обладнання – 0x29 (7 біт) ;
- датчик температури навколишнього середовища – від 0,038 до 50 клк;
- розрядність вихідного коду – 16 [105].

Модуль датчика температури й вологості DHT22

Модуль датчика температури й вологості підвищеної точності DHT22 містить усі необхідні додаткові компоненти для підключення датчика до мікроконтролера. Підходить для підключення як до контролерів Arduino так і до інших мікроконтролерів з напругою логічних рівнів 5 В та 3,3 В. Для зменшення споживаного струму модулем необхідно відключити світлодіод індикатора напруги живлення на платі модуля. Датчик модуля вимірює вологість і температуру точніше датчика вологості й температури DHT11 [233]. Загальний вид представлено на рисунку 2.73.



Рисунок 2.73 – Загальний вид датчика температури й вологості DHT22 [233]

Основні характеристики датчика температури й вологості DHT22 [234]:

1) температура:

- дозвіл виміру: 0.1°C;
- точність виміру : ± 0.5 ;
- вимірюваний діапазон : -40°C ~ 80°C;

2) вологість:

- дозвіл виміру: 0.1%RH;
 - точність виміру: $\pm 2\%RH$ (25°C);
 - вимірюваний діапазон: 0%RH ~ 99.9%RH
- 3) робоча напруга: від 3,3 до 5,5 В;
- 4) рекомендовані умови зберігання:
- температура: 10°C ~ 40°C;
 - вологість: 60%RH або нижче [106].

Модуль датчика струму на ACS709 від -75А до +75А.

Модуль датчика струму ACS709 є лінійним струмовим датчиком на мікросхемі Allegro ACS709LLFTR-35BB-T $\pm 75A$. Принцип роботи заснований на ефекті Холу з аналоговим виходом з низькоомним внутрішнім опором ($\sim 1,1$ мОм) і електричною ізоляцією до 2,1 кВ RMS. Датчик оптимізований для точного виміру струмів від -37,5 до 37,5 А, а аналоговий вихід напруги лінійний для значень струму до 75 А. Вихідну напругу центровано по $VCC / 2$ і має типову похибку $\pm 2\%$. Датчик працює в діапазоні споживаної напруги від 3 до 5,5 В, тому він може прямо взаємодіяти із системами 3,3 В та 5В (рис. 2.74) [235, 236].



Рисунок 2.74 – Загальний вид датчика струму ACS709 [237]

Ключові функції датчика струму на ACS709 [237]:

- оптимізована точність для двонаправленого вхідного струму від -37,5 до 37,5 А з лінійним діапазоном чутливості від -75 до 75 А;
- внутрішній опір провідного шляху зазвичай становить 1,1 мОм, а друкована плата виконана зі стовщеної міді, тому на платі виділяється дуже мало тепла;

- вбудована ізоляція значно зменшує ємнісний зв'язок від струмопроводу до вимірювальної матриці й запобігає зсуву дрейфу;
- використання датчика на ефекті Холу означає, що ІС здатна електрично ізолювати електроніку від датчика (до 2.1 кВ RMS), що дозволяє вставляти датчик у будь-якому місці провідника струму й використовувати в пристроях, що вимагають електричної ізоляції;
- смуга шириною 120 кГц, яка може бути додатково зменшена шляхом додавання конденсатора через штирі плати з написом «FILT»;
- висока точність і надійність: середня загальна помилка виходу 2% при кімнатній температурі із заводським калібруванням, надзвичайно стійка вихідна напруга спокою;
- робочий діапазон робочої температури від -40 до 150 °С;
- установлене користувачем граничне значення максимального струмового навантаження: вихідні дані виходу FAULT затримуються на низькому рівні, якщо струм перевищує встановлений поріг протягом часу, який може бути встановлений шляхом додавання зовнішнього конденсатора.

Збільшений діапазон від -75 до 75 А датчика повинен бути обмежений піковими перехідними струмами. При тестуванні виявлено, що мікросхема датчика може витримувати струм 50 А протягом 20 секунд або 37,5 А протягом 150 секунд, перш ніж перевищити максимальну температуру в 150 °С. Тому, якщо не вжиті спеціальні заходи відводу тепла від мікросхеми датчика, рекомендовано обмежити безперервні струми до рівня 30 А. Навіть при низькому опорі датчика 1,1 мОм плата може нагріватися настільки, щоб досягти високих температур, тому для тривалої роботи з великими струмами слід вжити додаткові заходи по відводу тепла.

Розташування й призначення виводів модуля датчика представлено на рисунку 2.75.

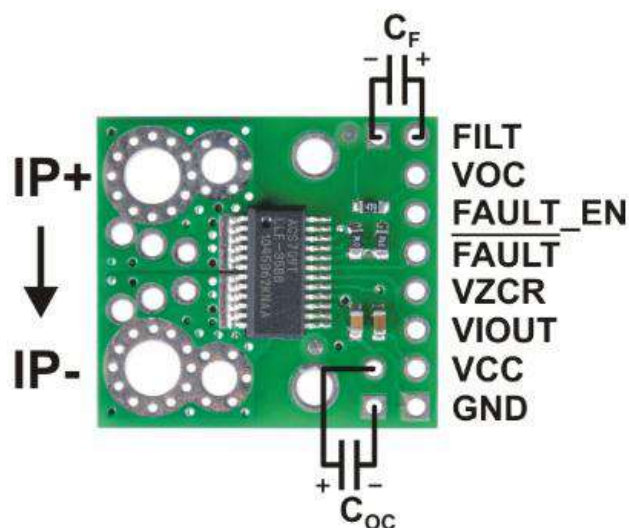


Рисунок 2.75 – Розташування й призначення висновків модуля датчика струму на ACS709

Єдиними підключеннями, які необхідні для використання цього датчика, є вхідний струм (IP + і IP-), напруга живлення (VCC і GND) і вихід датчика (VIOUT). Усі інші контакти є необов'язковими, як і два зовнішні конденсатори, показаних на рисунку 2.75 праворуч.

Датчик вимагає напруги живлення від 3 до 5,5 В для підключення через контакти VCC і GND, які позначені на платі шовкографією. Датчик виводить аналогову напругу, лінійно пропорційну вхідному струму. Вихідна напруга спокою становить $VCC / 2$ і змінюється на 28 мВ на ампер вхідного струму (при $VCC = 5\text{ В}$), при цьому позитивний струм збільшує вихідну напругу, а негативний струм зменшує вихідну напругу.

Для довільного вхідного струму вихідна напруга датчика може бути в більш загальному виді представлена як:

$$V_{IOUT} = (0,028\text{ В} / \text{А} \cdot i + 2,5\text{ В}) \cdot VCC / 5\text{ В}$$

Вивід VZCR є виходом опорної напруги й може використовуватися як опорний сигнал нульового струму. Він буде приблизно рівний $VCC/2$ і

дозволить більш точно обчислити вхідний струм від вихідної напруги VIOUT.

Вивід FILT дозволяє регулювати пропускну здатність плати, додаючи конденсатор CF до землі (для зручності доданий контакт заземлення поруч із контактом FILT) (рис. 2.76). Без якого-небудь зовнішнього фільтруючого конденсатора ширина смуги становить 120 кГц. У технічному паспорті міститься додаткова інформація про те, як конденсатор зовнішнього фільтра впливає на пропускну здатність.

Вивід FAULT перебуває в низькому рівні, коли струм перевищує поріг спрацьовування від перевантаження по струму несправності (I_{oc}). Ця точка перемикання задається напругою, що подається на вивід VOC, і залежить від ділянки напруги, показаного на принциповій схемі (рис. 2.76). За замовчуванням I_{oc} встановлюється на значенні струму в 57 А, але його можна змінити, додавши зовнішні резистори до ділянки напруги, щоб змінити напругу VOC. Після того, як контакт FAULT зафіксований на низькому рівні, його можна скинути, якщо вхідний сигнал FAULT_EN низький (цей вхід за замовчуванням піднятий на платі). Зовнішній конденсатор COS, може бути доданий для збільшення часу відгуку по перевантаженню по струму. Без цього конденсатора час відгуку, як правило, менший за 2 мкс.

Вхідний струм можна підключати до плати різними способами. Отвори із кроком 0,1", 3,5 мм і 5 мм доступні, як показано на рис. 2.75, для підключення штирів або клемних колодок. Для потужнострумових застосувань можна припаяти провідники безпосередньо до наскрізних отворів, або можна використовувати безконтактні клемні колодки, як показано на рисунку 2.75. Більші наскрізні отвори достатні для гвинтів М6.

Принципова схема модуля датчика струму на ACS709 представлена на рисунку 2.76.

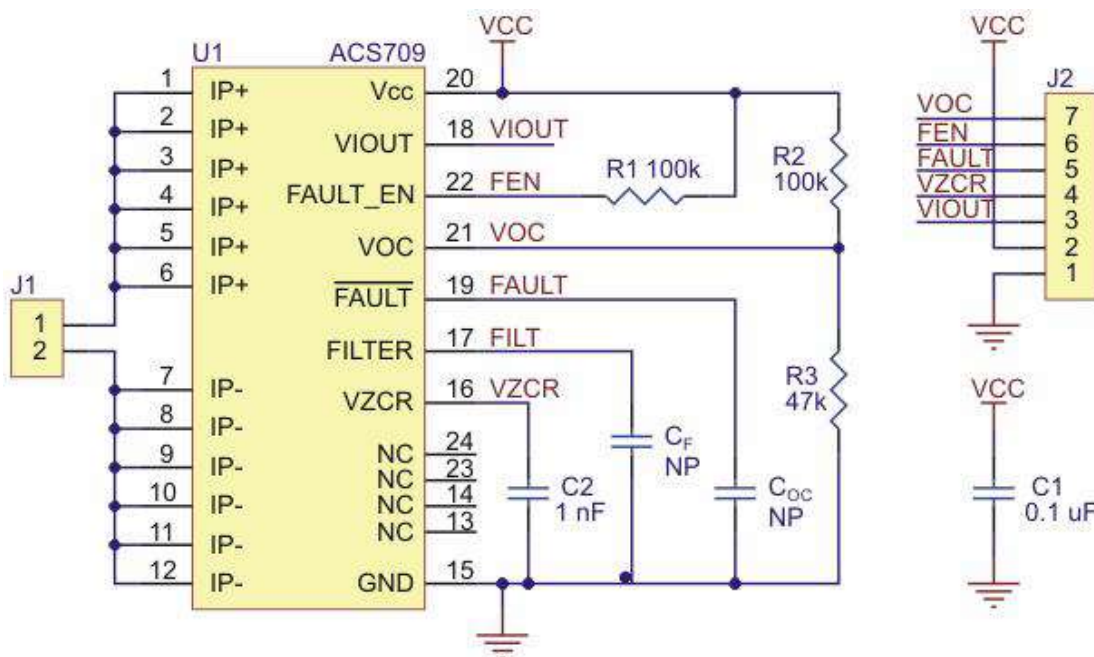


Рисунок 2.76 – Принципова схема модуля датчика струму на ACS709

Загальні характеристики модуля датчика струму на ACS709:

- мікросхема датчика: ACS709LLFTR-35BB-T;
- чутливість датчика: 0,018 В/А (для напруги живлення 3,3В); 0,028 В/А (для напруги живлення 5В);
- мінімальна логічна напруга: 3 В;
- максимальна логічна напруга: 5,5 В;
- споживаний струм: 11 мА;
- робочий діапазон робочої температури: від -40 до 150 °С;
- розміри: 23 x 21 мм [107].

Модуль лазерного датчика виявлення перешкоди

Модуль лазерного датчика виявлення перешкоди. Видає цифровий сигнал при потраплянні предмета в поле зору датчика. Дальність виявлення перешкоди становить до 1500 мм. Датчик відрізняється високою надійністю й простий у підключенні й використанні [238]. Загальний вид представлено на рисунку 2.77.

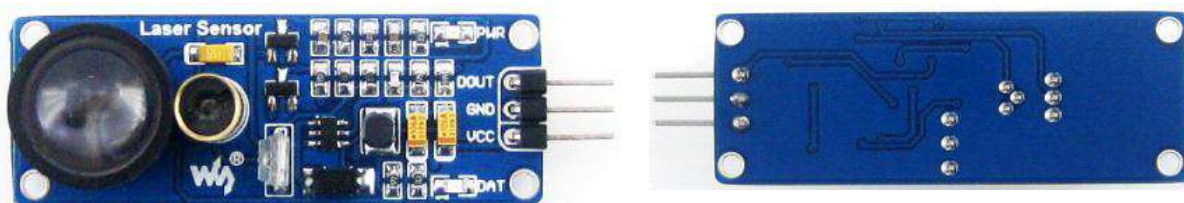


Рисунок 2.77 – Загальний вид модуля лазерного датчика перешкоди [238]

Загальні характеристики модуля лазерного датчика перешкоди [239]:

- відстань виявлення: від 0,8 до 1,5 м;
- напруга живлення: від 2,5 до 5,0 В;
- розміри: 47,7 мм * 17,9 мм;
- діаметр монтажного отвору: 2,0 мм [108].

H206-модуль оптичного датчика швидкості обертання

Може використовуватися для виміру швидкості обертання за допомогою диска із прорізами або як датчик перешкоди. Принцип – визначення об'єкта (перешкоди) між прорізами. Так само цей модуль можна використовувати в якості високоточного кінцевого датчика положення (ENDSTOP). На платі встановлений граничний компаратор напруги для одержання вихідного сигналу із крутими фронтами. Від інших аналогічних датчиків відрізняється компактними розмірами [240]. Загальний вид оптичного датчика представлено на рисунку 2.78.



Рисунок 2.78 – Загальний вид оптичного датчика [241]

Основні характеристики оптичного датчика [240]:

- напруга живлення 3,3 – 5 В;

- ширина паза датчика – 6 мм;
- тип виходу – аналоговий і цифровий;
- індикатор стану виходу.

Модуль камери Raspi CAM v2 для Raspberry Pi

Raspi CAM v2 камера с 8MP сенсором від Sony IMX219, поліпшеним фіксованим фокусом. Камера дозволяє робити статичні фото з роздільною здатністю 3280 x 2464 пікселів, знімає відео в трьох режимах — 1080p @ 30fps, 720p @ 60 fps і 640 x 480p @ 90fps [242]. Загальний вид камери Raspi CAM v2 представлено на рисунку 2.79.

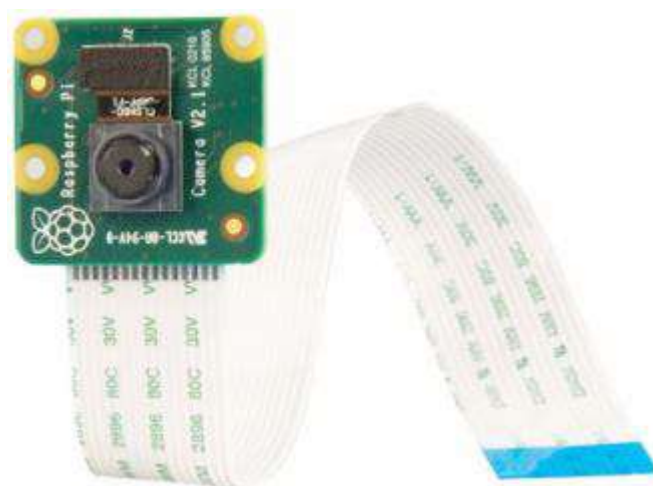


Рисунок 2.79 – Камера для Raspberry Pi - Raspi CAM v2. [242]

Основні характеристик Raspi CAM v2 [243]:

- 8-ми мегапіксельний сенсор Sony IMX219;
- поліпшений фіксований фокус;
- дозвіл фото: 3280 x 2464 пікселів;
- підтримувані формати відео: 1080p @ 30fps, 720p @ 60 fps і 640 x 480p @ 90fps;
- розмір матриці CCD : ¼ дюйма;

- $1.4\mu\text{m} \times 1.4\mu\text{m}$ пікселів з технологією Omnibsi для високої продуктивності (висока чутливість, низький рівень перехресних перешкод, низький рівень шуму);
- розміри: 25 мм * 23 мм * 9 мм [110].

3 АРХІТЕКТУРА ОДНОПЛАТНОГО КОМП'ЮТЕРА RASPBERRY PI 3 MODEL B PLUS

3.1 Основні елементи Raspberry Pi 3 model B plus

Raspberry Pi 3 Model B+ – одноплатний комп'ютер від Raspberry Pi Foundation, який працює на базі оновленого 4-ядерного 64-бітного SoC Broadcom BCM2837B0 та збільшеною тактовою частотою 1.4GHz. Бездротовий модуль також отримав оновлення: Wi-Fi став дводіапазонним стандарту IEEE 802.11ac, а Bluetooth – 4.2 BLE. "Малинка" має Gigabit Ethernet, що працює через USB 2.0 шину, що забезпечить швидкість передачі даних до 300Mbps. Незмінними залишилися обсяг оперативної пам'яті – 1GB RAM, формфактор та розташування всіх портів, що забезпечує сумісність більшості існуючих аксесуарів [244].

Raspberry Pi 3 Model B+, як і її попередники, відрізняється високим рівнем надійності, простотою налаштування та високою якістю виконання. Загальний вигляд і призначення основних елементів представлені на рис. 3.1.

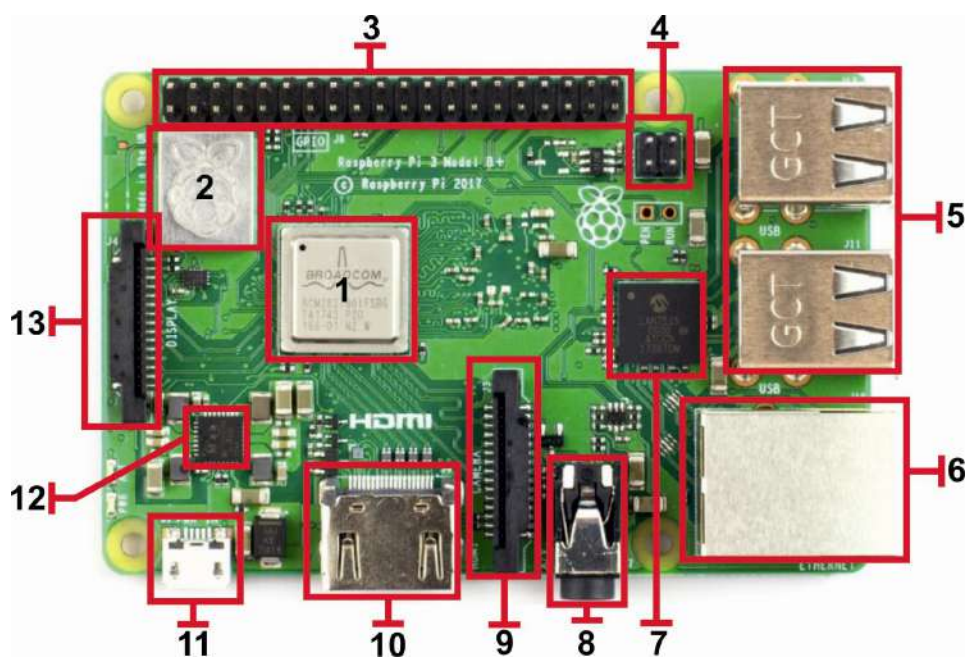


Рисунок 3.1 – Розташування основних елементів на платі
Raspberry Pi 3 Model B Plus [245]

Основні елементи на рисунку 3.1:

1 – чип Broadcom BCM2837B0, Cortex-A53, 64-bit, quad-core Soc @ 1.4Ghz with 1GB LPDDR2 SDRAM (поліпшене керування температурою та продуктивністю);

2 – чип Cypress CYW43455, IEEE 802.11 b/g/n/ac 2.4Ghz/5Ghz Wireless LAN (WLAN) ;

3 – Extended 40-pin GPIO header;

4 – Power over Ethernet (PoE) header (requires separate PoE HAT);

5 – 4xUSB 2.0 ports and Faster Ethernet over USB 2.0 (maximum throughput 300Mbps);

6 – роз'єм для конектора RJ45 (кручена пара);

7 – чип Microchip LAN7515;

8 – роз'єм 3,5 mm (jack), 4 pole stereo output and composite video port;

9 – MIPI CSI camera port;

10 – Full size HDMI;

11 – microUSB power connector 5V/2.5 A;

12 – HAT мікросхема;

13 – MIPI DSI display port.

Raspberry Pi 3 Model B + підтримує наступне периферійне обладнання та протоколи [111]:

– 48x GPIO;

– 2x I²C;

– 2x SPI;

– 2x UART;

– 2x SD/SDIO_M

– 1x HDMI 1.3a;

– 1x USB2 HOST/OTG;

– 1x DPI (Parallel RGB Display);

– 1x NAND interface (SMI);

– 1x 4-lane CSI Camera Interface (up to 1Gbps per lane);

2.2 Електричні характеристики Raspberry Pi 3 Model B +

Одноплатний міні-комп'ютер Raspberry Pi 3 Model B + має шість окремих джерел живлення. Всі вони повинні працювати постійно й не можна залишити жодного з них без живлення, навіть якщо певний інтерфейс або банк GPIO не використовується [247]:

- VBAT is used to power the BCM2837 processor core. It feeds the SMPS that generates the chip core voltage;
- 3V3 powers various BCM2837 PHYs, IO and the eMMC Flash;
- 1V8 powers various BCM2837 PHYs, IO and SDRAM;
- VDAC powers the composite (TV-out) DAC;
- GPIO0-27 VREF powers the GPIO 0-27 IO bank;
- GPIO28-45 VREF powers the GPIO 28-45 IO bank.

Робочий діапазон живлення Raspberry Pi 3 Model B + представлено в таблиці 3.1

Таблиця 3.1 – Робочий діапазон живлення Raspberry Pi 3 Model B +

Supply	Description	Minimum	Typical	Maximum	Unit
VBAT	Core SMPS Supply	2.5	-	5.0 + 5%	V
3V3	3V3 Supply Voltage	3.3 - 5%	3.3	3.3 + 5%	
1V8	1V8 Supply Voltage	1.8 - 5%	1.8	1.8 + 5%	
VDAC	TV DAC Supply ^α	2.5 - 5%	2.8	3.3 + 5%	
GPIO0-27 VDD	GPIO0-27 I/O Supply Voltage	1.8 - 5%	-	3.3 + 5%	
GPIO28-45 VDD	GPIO28-45 I/O Supply Voltage	1.8 - 5%	-	3.3 + 5%	
SDX VDD	Primary SD/eMMC Supply Voltage	1.8 - 5%	-	3.3 + 5%	

^α– Requires a clean 2.5-2.8V supply if TV DAC is used, else connect to 3V3

Основні електричні характеристики Raspberry Pi 3 Model B + наведено в таблиці 3.2.

Таблиця 3.2 – Основні електричні характеристики

Symbol	Parameter	Minimum	Maximum	Unit
VBAT	Core SMPS Supply	-0.5	6.0	V
3V3	3V3 Supply Voltage		4.10	
1V8	1V8 Supply Voltage		2.10	
VDAC	TV DAC Supply		4.10	
GPIO0-27 VDD	GPIO0-27 I/O Supply Voltage			
GPIO28-45 VDD	GPIO28-45 I/O Supply Voltage			
SDX VDD	Primary SD/eMMC Supply Voltage			

Напруги вище перераховані у таблицях 3.1–3.2 можуть привести до необоротного ушкодження Raspberry Pi 3 Model B +. Це тільки рейтинг максимального стресу, який не впливає на функціонування пристрою в тих або інших умовах.

Основні характеристики Raspberry Pi 3 Model B Plus по постійному струму (DC) представлено в таблиці 3.3.

Таблиця 3.3 – DC Characteristics

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
1	2	3	4	5	6	7
V _{IL}	Input low voltage ^α	VDD IO = 1.8V	-	-	0.6	V
		VDD IO = 2.7V	-	-	0.8	

		VDD IO = 3.3V	-	-	0.9	
V _{IH}	Input high voltage ^α	VDD IO = 1.8V	1.0	-	-	V
		VDD IO = 2.7V	1.3	-	-	
		VDD IO = 3.3V	1.6	-	-	
I _{IL}	Input leakage current ^α	TA = +85C	-	-	5	μA
C _{IN}	Input capacitance	-	-	5	-	pF
		VDD IO = 1.8V, IOL = -2mA	-	-	0.2	
V _{OL}	Output low voltage ^β	VDD IO = 2.7V, IOL = -2mA	-	-	0.15	V
		VDD IO = 3.3V, IOL = -2mA	-	-	0.14	
		VDD IO = 1.8V, IOH = 2mA	1.6	-	-	
V _{OH}	Output high voltage ^β	VDD IO = 2.7V, IOH = 2mA	2.5	-	-	V
		VDD IO = 3.3V, IOH = 2mA	3.0	-	-	
		VDD IO = 1.8V, VO = 0.4V	12	-	-	
I _{OL}	Output low current ^ξ	VDD IO = 2.7V, VO = 0.4V	17	-	-	mA
		VDD IO = 3.3V, VO = 0.4V	18	-	-	
		VDD IO = 1.8V, VO = 1.4V	10	-	-	
I _{OH}	Output high current ^ξ	VDD IO = 2.7V, VO = 2.3V	16	-	-	mA

		VDD IO = 3.3V, VO = 2.3V	17	-	-	
R _{PU}	Pullup resistor	-	50	-	65	kΩ
R _{PD}	Pulldown resistor	-	50	-	65	

^α – Hysteresis enabled;

^β – Default drive strength (8mA)

^ξ – Maximum drive strength (16mA).

Основні характеристики цифрових входів / виходів по змінному струму наведено в таблиці 3.4

Таблиця 3.4 – Характеристики цифрових входів/виходів змінного струму

Pin Name	Symbol	Parameter	Minimum	Typical	Maximum	Unit
Digital outputs	t_{rise}	10-90% rise time ^α	-	1.6	-	ns
Digital outputs	t_{fall}	90-10% fall time ^α	-	1.7	-	
GPCLK	t_{JOSC}	Oscillator-derived GPCLK cycle-cycle jitter (RMS)	-	-	20	ps
GPCLK	t_{JPLL}	PLL-derived GPCLK cycle-cycle jitter (RMS)	-	-	48	

^α – Default drive strength, CL = 5pF, VDD IOx = 3.3V

Загальний вид цифровий сигнал входу/виходу представлено на рисунку 3.3



Рисунок 3.3 – Цифровий сигнал входу/виходу

3.3 General Purpose Input Output (GPIO)

GPIO (General Purpose Input Output) – це низькорівневий інтерфейс ведення-вивід прямого керування який дозволяє підключати різні модулі й датчики (рисунок 3.1 поз. 3). Raspberry Pi 3 має на борті 40-пинову рейку GPIO (рис. 3.4), але так як 12 з них представляють із себе піни живлення 3.3 В, 5 В і загальні піни GND [248].

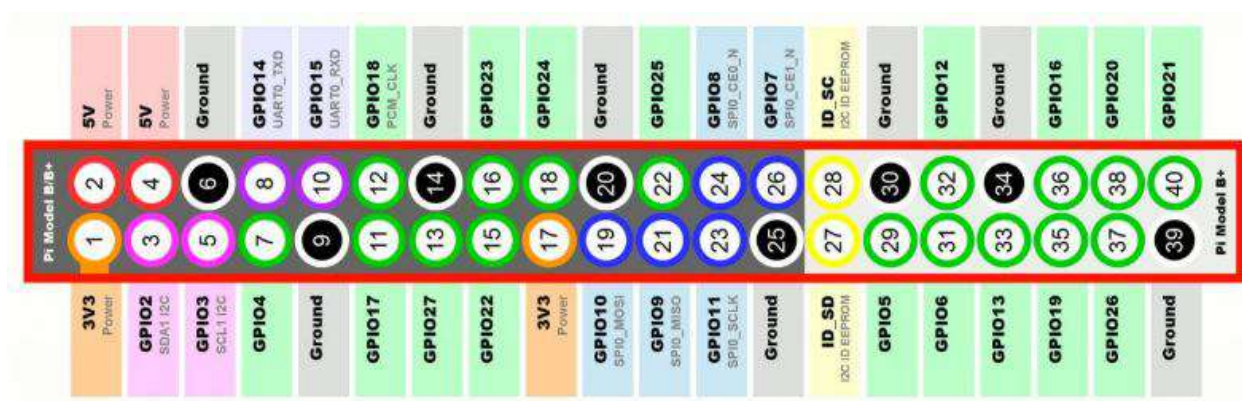


Рисунок 3.4 – Призначення пінів роз'єму GPIO на Raspberry Pi 3

Також 27 (BCM 0) і 28 (BCM 1) піни використовуються для конфігурації EEPROM Raspberry Pi 3 для роботи з Нат-обладнаннями (Hardware Attached on Top – обладнання поверхневого монтажу, по суті – звичайні плати розширення) і використання цих пінів не рекомендується. Проте вони є повноцінними GPIO-пінами. Для запобігання проблем, при написанні програм, необхідно явно вказувати який режим нумерації буде використаний.

Існує кілька варіантів нумерації пінів Raspberry Pi 3: Board (фізична нумерація один по одному) і BCM (нумерація із чипа). Також з бібліотекою WiringPi використовується своя нумерація (рис. 3.5).

WiringPi					
<i>BCM</i>					
Board					
3,3 V Power		1	2	5 V Power	
WiringPi 8	<i>BCM 2</i>	3	4	5 V Power	
WiringPi 9	<i>BCM 3</i>	5	6	GND	
WiringPi 7	<i>BCM 4</i>	7	8	<i>BCM 14</i>	WiringPi 15
GND		9	10	<i>BCM 15</i>	WiringPi 16
WiringPi 0	<i>BCM 17</i>	11	12	<i>BCM 18</i>	WiringPi 1
WiringPi 2	<i>BCM 27</i>	13	14	GND	
WiringPi 3	<i>BCM 22</i>	15	16	<i>BCM 23</i>	WiringPi 4
3,3 V Power		17	18	<i>BCM 24</i>	WiringPi 5
WiringPi 12	<i>BCM 10</i>	19	20	GND	
WiringPi 13	<i>BCM 9</i>	21	22	<i>BCM 25</i>	WiringPi 6
WiringPi 14	<i>BCM 11</i>	23	24	<i>BCM 8</i>	WiringPi 10
GND		25	26	<i>BCM 7</i>	WiringPi 11
WiringPi 30	<i>BCM 0</i>	27	28	<i>BCM 1</i>	WiringPi 31
WiringPi 21	<i>BCM 5</i>	29	30	GND	
WiringPi 22	<i>BCM 6</i>	31	32	<i>BCM 12</i>	WiringPi 26
WiringPi 23	<i>BCM 13</i>	33	34	GND	
WiringPi 24	<i>BCM 19</i>	35	36	<i>BCM 16</i>	WiringPi 27
WiringPi 25	<i>BCM 26</i>	37	38	<i>BCM 20</i>	WiringPi 28
GND		39	40	<i>BCM 21</i>	WiringPi 29

Рисунок 3.5 – Варіант нумерації пінів Raspberry Pi 3: Board и BCM

Кожний з 28 пінів може бути встановлений як у режим цифрового виходу OUTPUT, так і в режим цифрового входу INPUT, як показано на рисунку 3.6.

Максимальний вихідний струм кожного піна не повинен перевищувати 16 мА.

$$\max I_{out} = 16mA$$

Сумарний вихідний струм усіх пінів не повинен перевищувати 50 мА

$$\max \sum I_{out} = 50mA$$

5-вольтові піни можуть давати більший струм, який залишається після живлення Raspberry Pi 3 і інших периферійних обладнань (клавіатури, миші) – до 500 мА.

$$\max \sum I_{5v-out} = 500mA$$

За замовчуванням усі піни, крім BCM14 і BCM15, перебувають у режимі INPUT, причому піни BCM0-BCM8 і BCM15 підтягнуті до одиниці резисторами, що підтягують (pullup) (рис. 3.6). Саме із цієї причини мультиметр покаже напругу на цих пінах. Інші піни стягнуті до нуля. Кожен з 28 пінів має резистори, що підтягують (pullup) і зкидають (pulldown), завдяки чому, у режимі INPUT може бути підтягнутий до логічної одиниці, або зкинутий до нуля.

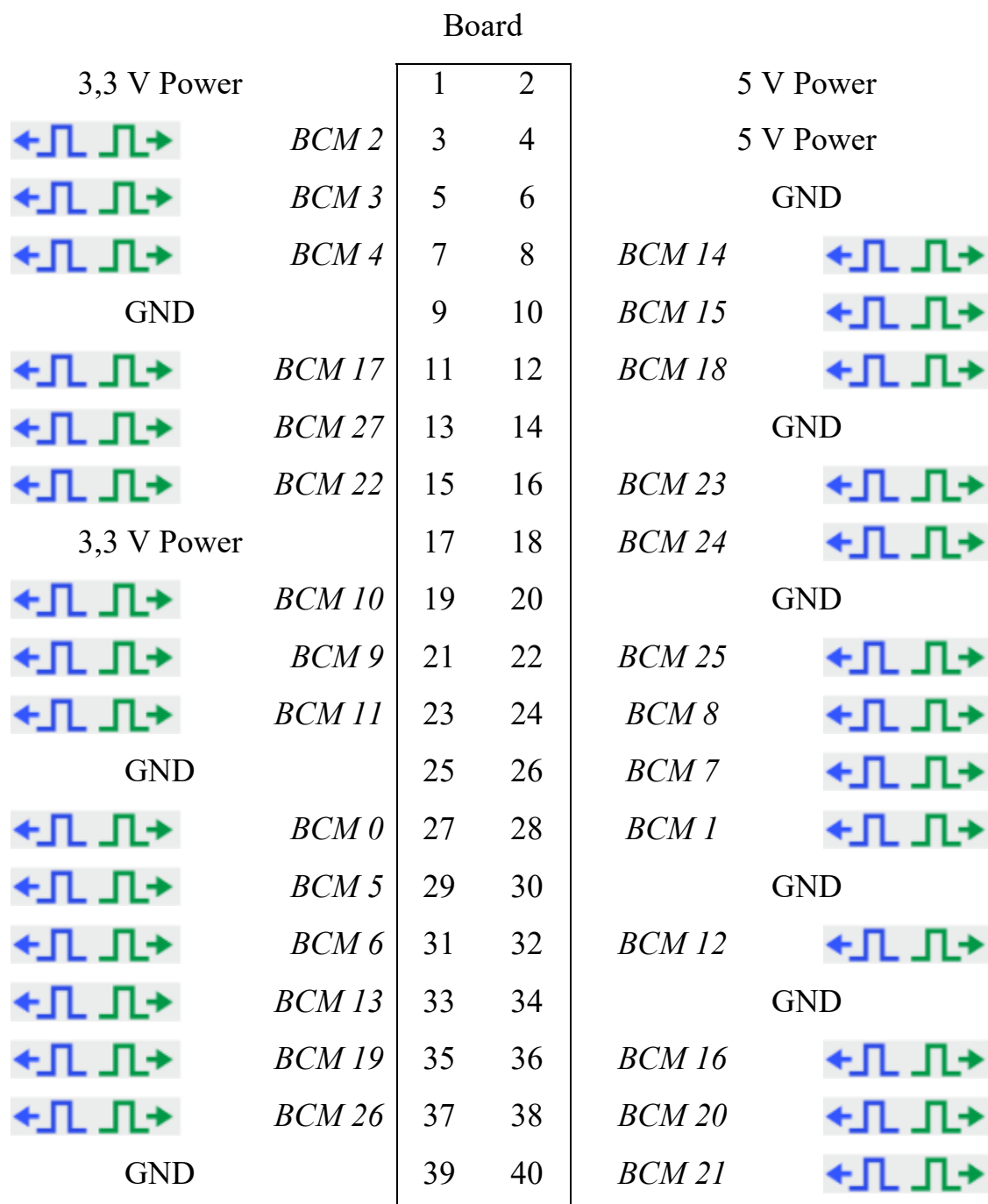


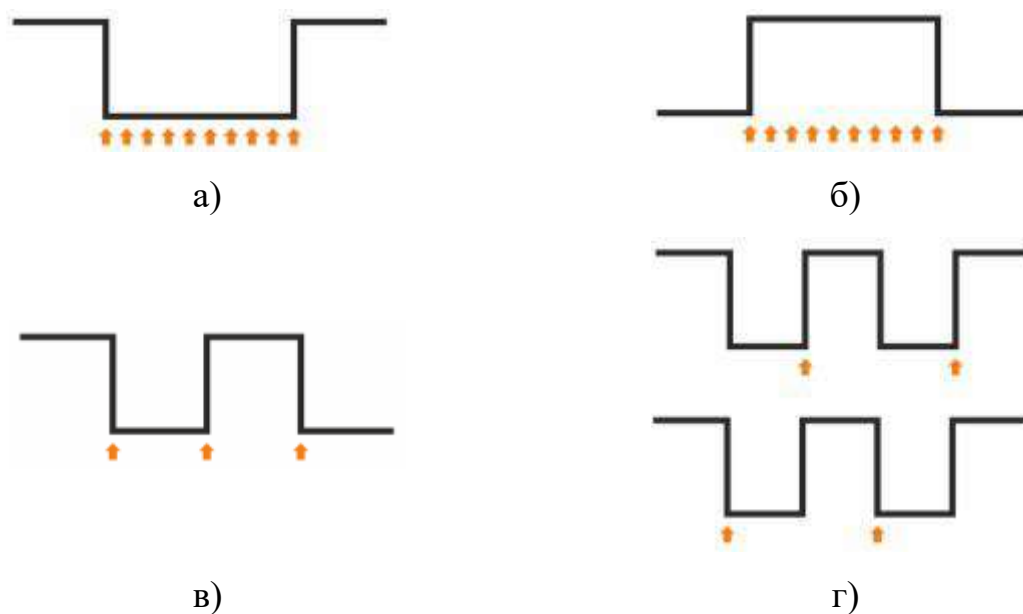
Рисунок 3.6 – Режими роботи роз'ємів GPIO

Номінали опорів не постійні й рівні:

- для резистора, що підтягує 50 КОм – 65 КОм;
- для зкидаючого резистора 50 КОм – 60 Ком.

Номінал опору підтягуючого / зкидаючого резистора для пінів BCM2 і BCM3 – 1.8 КОм.

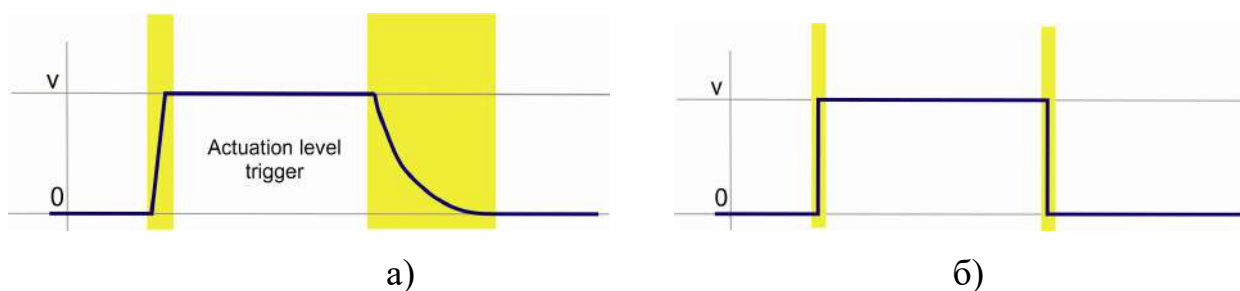
Кожен з 28 пінів у режимі INPUT може генерувати переривання: по спаду, по фронту, по одиниці, по нулю, по зміні сигналу, а також в асинхронному режимі: по фронту та по спаду, як показано на рисунку 3.7.



a) low; б) high; в) change; г) async

Рисунок 3.7 – Режими роботи пінів у стані INPUT

Також усі піни в режимі INPUT мають вхідну фільтрацію на тригері Шмітта (перетворюють аналоговий сигнал у цифровий з різкими переходами між станами), як показано на рисунку 3.8.



a) вхідний реальний сигнал до тригера Шмітта;

б) вхідний сигнал після тригера Шмітта

Рисунок 3.8 – Приклад фільтрації на тригері Шмітта

Raspberry Pi 3 не має аналогових входів/виходів. Для реалізації подібного функціонала потрібно використовувати зовнішні АЦП/ЦАП, наприклад, АЦП ADS1115 (I²C) або АЦП MCP3008.

3.4 Послідовний інтерфейс I²C

Інтерфейс I²C (I2C або TWI) був розроблений компанією Philips для внутрішнього зв'язку периферії в побутовій електроніці. Для передачі даних і адресації використовуються всього два дроти. При першому, швидкому погляді на структуру кадру й опис в Datasheet, видається що інтерфейс I²C дуже складний і відразу в структурі його кадру не розібратися. Але саме це і є його перевагою перед іншими інтерфейсами [249, 250].

У порівнянні з інтерфейсом SPI, в I²C є такі позитивні моменти [251]:

- більш наочна структура кадру;
- можливість відстеження виникаючих помилок при прийманні та передачі;
- можливість використання переривання на вході даних по шині I²C, у той час як мікроконтролер перебуває в сплячому режимі;
- для вибору обладнання не потрібно задіяти додаткові виводи мікроконтролера;
- малошвидкісне обладнання самостійно знижує швидкість шини.

Але наряду з перевагами необхідно вказати на недоліки:

- низька швидкість передачі даних, до 400 Кбіт/с (а якщо врахувати ще, що спочатку передається адреса обладнання, то реальна швидкість ще нижче);
- обмеження по числу однакових пристроїв, що адресуються (звичайно виробники периферії із шиною I²C залишають 3 біта адреси обладнання на вибір користувача, а значить максимум можна використовувати 8 однакових одиниць обладнання).

До двопровідної шини I^2C всі пристрої підключаються паралельно й обидва дроти підтягуються до шини живлення. По лінії SDA передаються дані, а по SCL здійснюється тактування. Приклад підключення по шині I^2C представлено на рисунку 3.9.

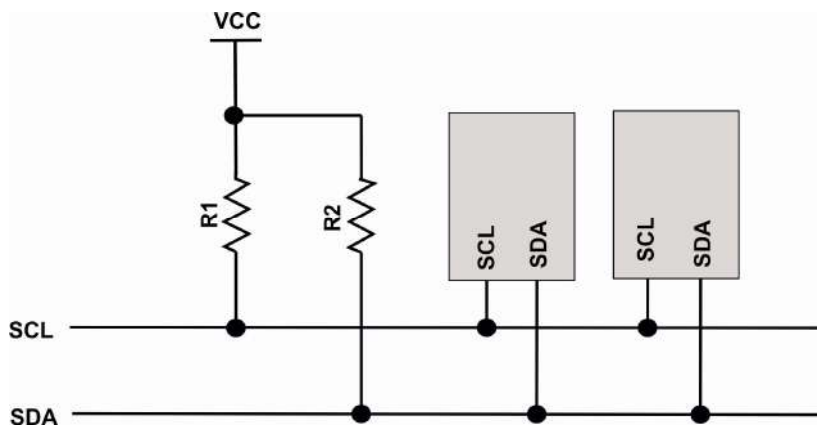


Рисунок 3.9 – Схема підключення резисторів і обладнання до шини I^2C [251]

Номінал резисторів, залежить від довжини шини, а також від кількості підключеного обладнання. Якщо використовувати пару пристроїв, що підключаються до шини, не враховуючи мікроконтролер, і довжину лінії до 30 см, то можна використовувати резистори номіналом 10 кОм.

Щоб почати передачу Raspberі Pi повинен послати на шину I^2C сигнал "START" – спочатку низький рівень установлюється на лінії SDA, а потім на лінії SCL. Далі відбувається передача адреси та даних. По закінченню передачі на шину посилає сигнал "STOP" – спочатку високий рівень установлюється на шині SCL, а потім на шині SDA. Форма сигналу "START" і "STOP" представлено на рисунку 3.10.

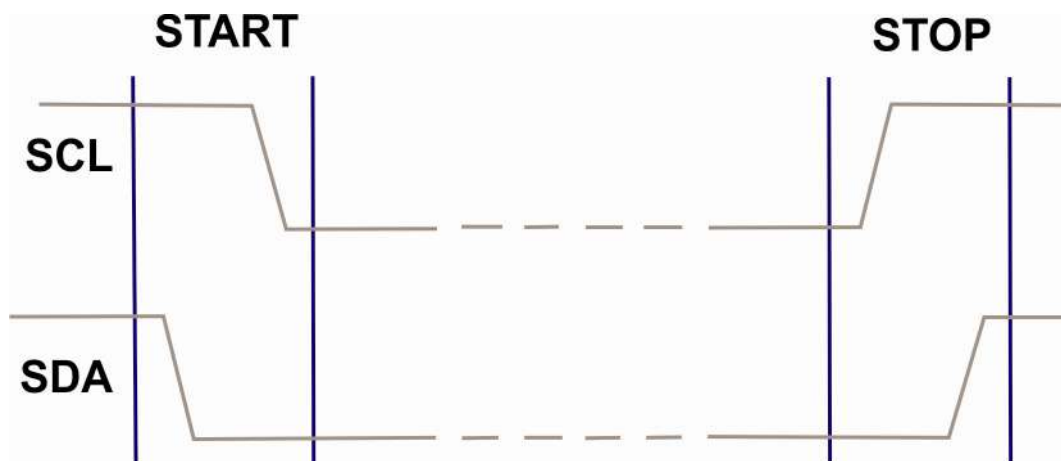
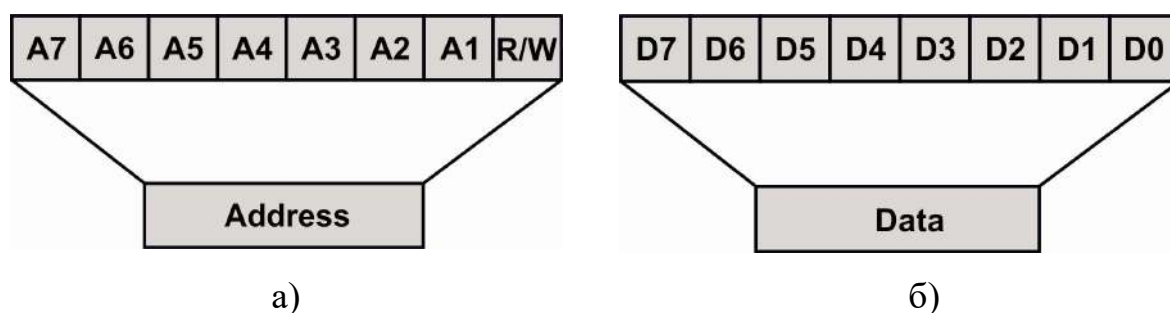


Рисунок 3.10 – Сигнали "START", "STOP"

Адреса й дані передаються пакетами по 8 біт. У пакеті адреси 1 біт використовується для інформування веденого пристрою: зчитування (1) даних або ж їх запис (0). На рисунку 3.11 представлений формат пакету адрес і даних.



а) фрагмент пакету адрес; б) фрагмент пакету даних

Рисунок 3.11 – Формат пакетів адрес і даних

До кожного пакету додається ще один біт – підтвердження (ACK) або не підтвердження (NACK). При передачі адреси, а також запису даних цей біт передається веденому пристрою. При читанні – провідному (Raspberry Pi).

У режимі **Master Write** (рис. 3.12) мікроконтролер, після команди "START", передає адресу обладнання, у яке прагне записати дані (установлюючи біт читання / запису в 0). Якщо на шині присутнє

обладнання з такою адресою, то воно посилає відповідь "ACK" і далі відбувається пересилання даних від провідного обладнання. Якщо обладнання з такою адресою немає, то передача закінчується. Після закінчення передачі даних провідне обладнання посилає в шину команду "STOP".

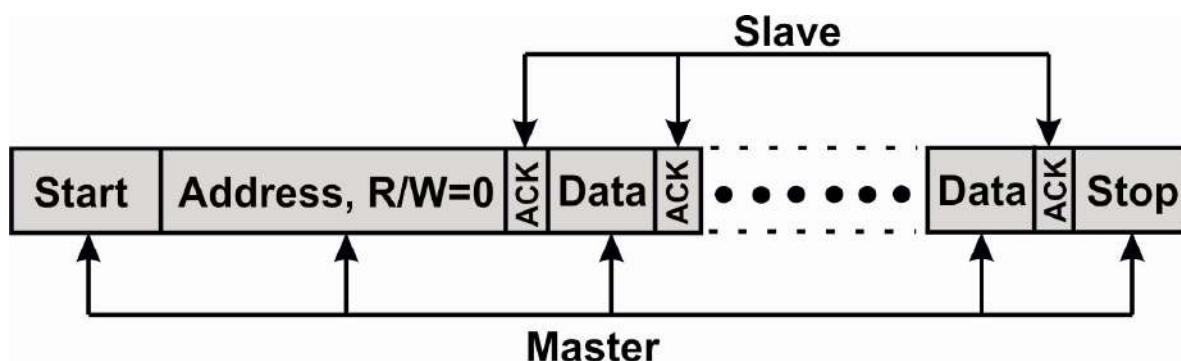


Рисунок 3.12 – Режим Master Write

У режимі *Master Read* (3.13) мікроконтролер, після команди "START", передає адресу обладнання, з якої прагне отримати дані (установлюючи біт читання / запису в 1). Якщо на шині присутнє обладнання з такою адресою, то воно посилає відповідь "ACK", і далі відбувається пересилання даних від веденого пристрою, яке підтверджується провідним пристроєм. Після прийняття останнього байта даних від веденого, провідний пристрій посилає відповідь "NACK". Якщо обладнання з такою адресою немає, то передача закінчується. Після закінчення передачі даних провідний пристрій посилає в шину команду "STOP".

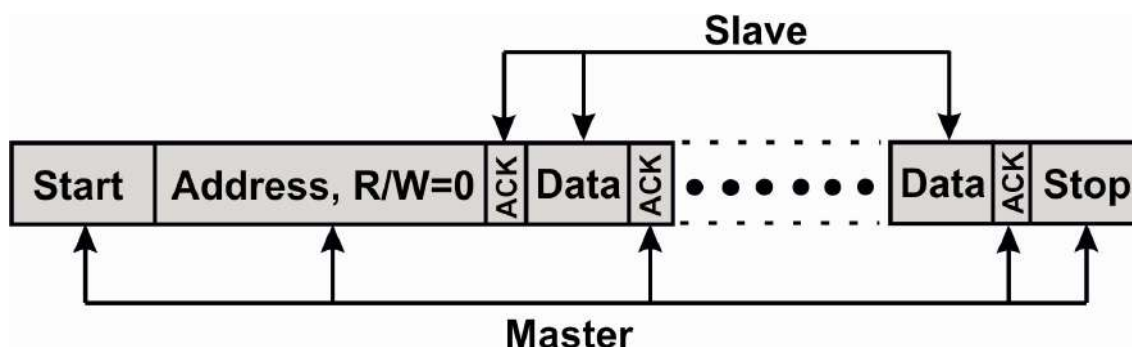


Рисунок 3.13 – Режим Master Read

Однією із цікавих особливостей передачі даних по протоколу I²C є здатність веденого пристрою знижувати швидкість передачі даних. Якщо швидкість висока, то ведений пристрій втримує лінію тактування (SCL) у низькому стані та відпускає її коли готове прийняти наступний біт.

3.5 Послідовний інтерфейс SPI

SPI (Serial Peripheral Interface) чи (послідовний периферійний інтерфейс) – це синхронна чотирипровідна шина. Вона являє собою з'єднання двох синхронних зсувних регістрів, які є центральним елементом будь-якого SPI обладнання. Для з'єднання використовується конфігурація провідний / ведений. Тільки провідний може генерувати імпульси синхронізації. У схемі завжди тільки один провідний, на відміну від тієї ж шини I²C, де можливий варіант із більш ніж одним ведучим. Кількість ведених може бути різною. У загальному випадку вихід ведучого з'єднується із входом веденого, і навпаки, вихід веденого з'єднується із входом ведучого. При подачі імпульсів синхронізації на вихід SCK, дані передаються ведучим з виходу MOSI, і отримуються веденим по входу MISO. У такий спосіб, якщо подати кількість імпульсів синхронізації, відповідно до розрядності здвигового регістру, то дані в регістрах поміняються місцями. Тому можна визначити, що SPI завжди працює в повнодуплексному режимі [252].

Контролер SPI, як правило, реалізується периферійним блоком в MCU або етпри. У більшості чипів він може працювати як у режимі провідного, так і в режимі веденого. Існує кілька способів підключення SPI обладнання, але в даному розділі ми розглянемо найпоширеніший «Каскадна схема» підключення SPI обладнання, який представлено на рисунку 3.14.

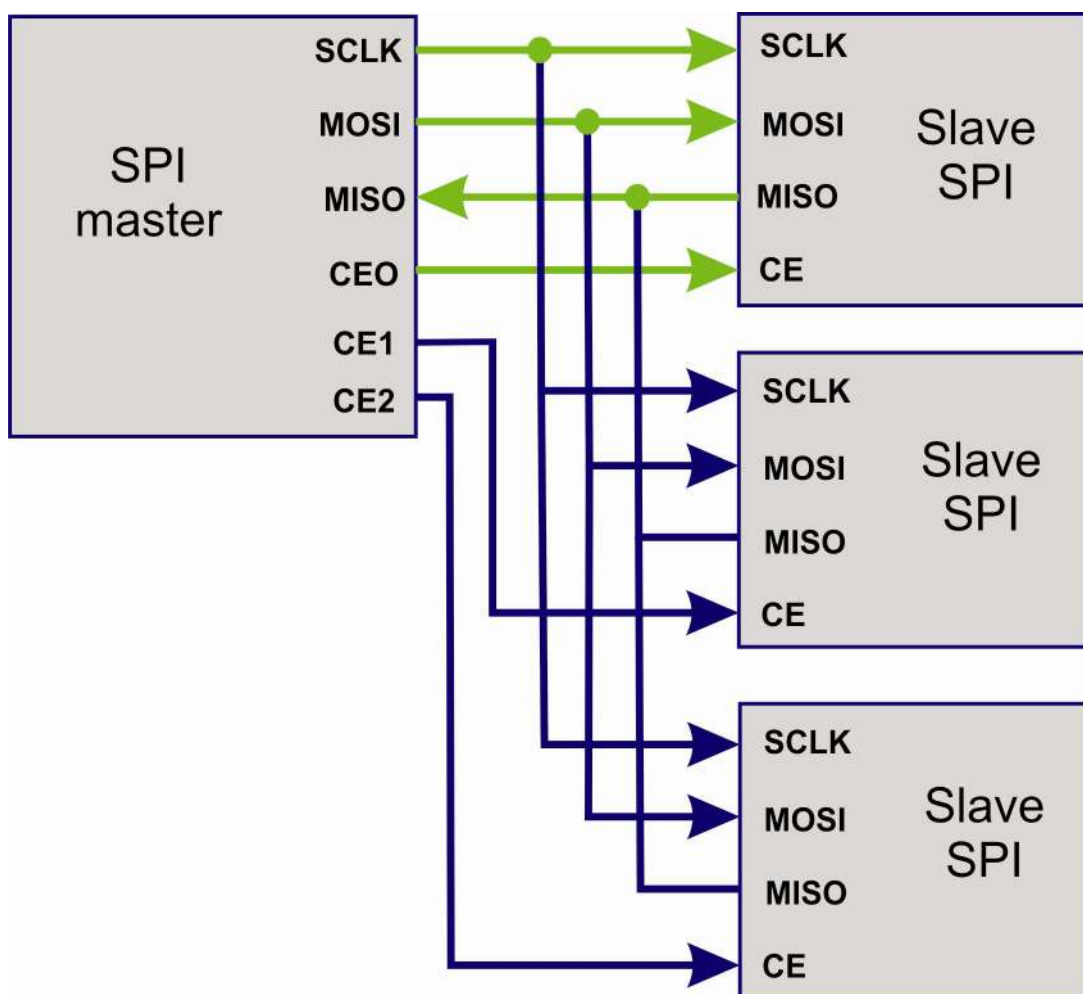


Рисунок 3.14 – Каскадна схема підключення SPI обладнання [252]

У цьому випадку до провідного пристрою всі ведені підключаються паралельно, за винятком сигналу вибору веденого ($\sim CE$). Для кожного веденого необхідний окремий сигнал вибору (на рис. 3.14 вони позначені як CE_n). Для сигналів вибору веденого можуть використовуватися як спеціально призначені для цього виходи Spi-контролера, так і порти введення/виводу загального призначення (GPIO) Raspberry Pi.

Два провідники використовуються для передачі даних, один для подачі тактових імпульсів і по одному сигналу для вибору провідним кожного з ведених. У каскадній схемі підключення SPI використовується наступний набір сигналів [253]:

MOSI – Master Output, Slave Input (вихід провідного, вхід веденого). Даний сигнал призначений для послідовної передачі даних від ведучого до веденого. Також може називатися SDO, DO і т.п.;

MISO – Master Input, Slave Output (вхід провідного, вихід веденого). Даний сигнал призначений для послідовної передачі даних від веденого до ведучого. Може називатися SDI, DI і т.п.;

SCK – Serial Clock (сигнал синхронізації). Використовується для синхронізації при передачі даних. Також може мати назву SCLK, CLK і ін.;

$\sim$ CE – Chip Select (вибір мікросхеми). За допомогою даного сигналу відбувається активація веденого пристрою. Звичайно він є інверсним, тобто низький рівень вважається активним. Іноді його називають $\sim$ SS (Slave Select, «вибір веденого»).

Окремим випадком незалежного підключення є варіант із одним єдиним веденим. У такому випадку, щоб обладнання завжди було в активному стані, можна підтягти сигнал $\sim$ CS до землі. Але робити це вкрай не рекомендується, тому що ведений пристрій може використовувати сигнал CS для ініціалізації або для інших службових цілей.

Існує чотири режими роботи SPI. Дані чотири режими являють собою комбінацію двох бітів:

CPOL (Clock Polarity) – визначає початковий рівень (полярність) сигналу синхронізації. CPOL=0 показує, що сигнал синхронізації починається з низького рівня, так що передній фронт є наростаючим, а задній – падаючим. CPOL=1, сигнал синхронізації починається з високого рівня, у такий спосіб передній фронт є падаючим, а задній – наростаючим;

CPHA (Clock Phase) – фаза синхронізації, визначає по якому із фронтів синхронізуючого сигналу робити вибірку даних. CPHA=0 показує що необхідно робити вибірку по передньому фронті, а CPHA=1 показує що вибірку даних необхідно робити по задньому фронті

Ці два біти й утворюють номер режиму. CPOL є старшим бітом, а CPHA – молодшим. Іноді в документації до обладнання явно не вказують

номер режиму, але його завжди можна легко визначити по тимчасових діаграмах.

3.6 Послідовний порт UART

UART (Universal Asynchronous Receiver/Transmitter – універсальний асинхронний приймач / передавач) – протокол послідовного зв'язку, у якому дані передаються послідовно, тобто по-бітно. При такому зв'язку за один раз передається байт даних [254-256].

Існує велика кількість різноманітних протоколів зв'язку, призначених для використання з UART. Найбільш відомі:

- MODBUS – сімейство протоколів типу «запит-відповідь», популярне в промисловій автоматизації. Орієнтовані на управління обладнанням за допомогою коротких команд [257,258];

- AT-команди телефонних модемів. Набір текстових команд, що дозволяють керувати роботою модему при встановленні з'єднання [259];

- PPP-протокол, який широко використовувався при підключеннях до інтернету через модем. Дозволяє виконувати аутентифікацію користувача у провайдера, шифрувати дані тощо [260, 261];

- IRDA – сімейство протоколів фізичного рівня для оптичного бездротового керування [262, 263].

Протокол UART використовує певну структуру кадрів для передачі байту даних. Структура асинхронного повідомлення наступна [264]:

- біт START – вказує, що послідовний зв'язок почався і цей біт завжди має низький рівень сигналу;

- пакет бітів даних – бітами даних можуть бути пакети від 5 до 9 біт. Як правило, використовується 8-бітний пакет даних, який завжди надсилається після START-біта;

– біт STOP – це, зазвичай, один або два біти. Відправляється після пакетів бітів даних, щоб вказати кінець кадру. Біт STOP завжди має високий логічний рівень [116].

На рисунку 3.15 представлена структура основного кадру протокол UART.

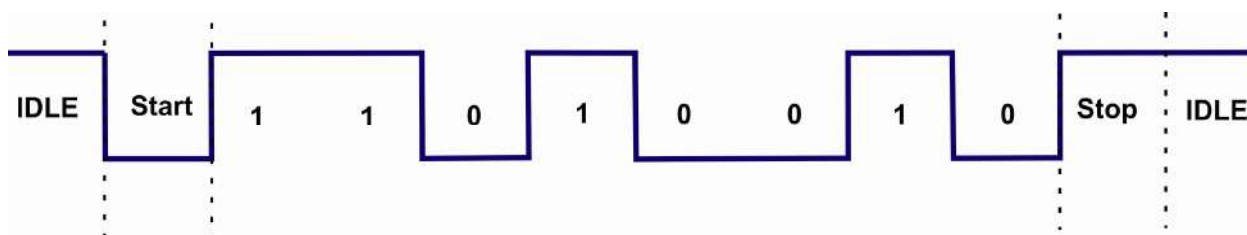


Рисунок 3.15 – Структура основного кадру протокол UART [264]

Кадр може також мати біт PARITY, який передається після STOP-біта. UART зазвичай використовується в Raspberry Pi (RPi), як зручний спосіб керування через GPIO, або для доступу до завантажувальних повідомлень ядра з послідовної консолі (включено за замовчуванням). Він також може бути використаний як спосіб для взаємодії з Arduino, ATmega, ESP8266 та інших мікроконтролерів з RPi.

Необхідно бути обережними з рівнями логіки між пристроями, бо, наприклад, RPi має 3,3 В, а Arduino – 5 В. Підключивши їх разом можна їх пошкодити.

Raspberry Pi має два вбудованих UART:

- PL011 UART [265];
- mini UART [266].

PL011 UART – це UART на базі ARM. Він має кращу пропускну здатність, ніж mini UART. Хоча RPi має два UART, але має лише одну пару виводів TXD і RXD для роботи.

UART PL011 – є основним UART для моделей без функції Bluetooth і пов'язаний безпосередньо з консоллю Linux. Це означає, що через цей UART можна надсилати команди Linux з ПК на RPi. З іншого боку, mini UART є

Linux консоллю UART для моделей з Bluetooth, такими як Raspberry Pi 3 і Raspberry Pi Zero W. Для цих моделей UART PL011 зв'язаний безпосередньо з модулем Bluetooth.

UART PL011 є більш надійним, ніж міні UART, оскільки останній має менші за обсягом FIFO, не може контролювати потік, а швидкість передачі даних залежить від тактової частоти CPU. Оскільки частота ядра графічного процесора змінюється, то змінюється і частота UART, що, в свою чергу, змінює швидкість передачі даних через UART. Це робить міні UART нестабільним, що може призвести до втрати даних або їх пошкодження. Щоб зробити міні UART стабільним, треба змінити основну частоту. Також міні UART не підтримує перевірку парності. Для кращого та ефективного зв'язку рекомендується використовувати PL011 UART замість міні UART.

4 ОПЕРАЦІЙНА СИСТЕМА RASPBIAN ДЛЯ РОБОТИ З OPENCV МООВОЮ PAYTEN

4.1 OS Raspbian за допомогою NOOBS

Raspberry Pi (RPi) — це одноплатний мікрокомп'ютер, у якого є всі ті ж ознаки, що й у звичайних персональних комп'ютерів і ноутбуків. До нього можна підключити монітор, клавіатуру, мишу, аудіо колонку, а також Інтернет-кабель. Як і персональний комп'ютер RPi працює під керуванням повноцінних операційних систем (ОС), таких як: Raspbian (Debian) [267], Android [268] і навіть Windows 10 (IoT) [269]. Операційна система в Rpi зберігається на карті пам'яті формату microSD. Там же зберігаються й усі файли користувача. Щоб почати роботу з Raspberry Pi, необхідно встановити на карту пам'яті операційну систему. Поділимо даний підрозділ на *основні кроки*.

Крок 1 – рекомендується використовувати карту пам'яті розміром не менш 8 Гб. Найпопулярнішою ОС для Raspbrery Pi вважається Raspbian. Це по суті модифікована Debian. Є два варіанти установки Raspbian на карту пам'яті:

- копіювання образу карти пам'яті, із наперед встановленим Raspbian, що робиться за допомогою Win32Diskimager;
- копіювання файлів інсталяції NOOBS, які в автоматичному режимі встановлять Raspbian або іншу ОС.

Крок 2 – завантаження архіву (<https://www.raspberrypi.org/downloads/noobs/>) з файлами образів NOOBS. Можна обрати NOOBS або NOOBS Lite [270, 271]. У першому випадку буде завантажено установник у комплекті з Raspbian. У другому випадку установник буде порожній, але можна буде «дозавантажити» будь-яку ОС із інтернету. Обираємо перший варіант.

Крок 3 – розпаковуємо завантажений архів.

Крок 4 – копіюємо всі файли з архіву на порожню карту пам'яті.

Крок 5 – запускаємо Raspberry Pi. Усі Raspberry Pi починають своє завантаження з райдужного екрану, як показано на рисунку 4.1.

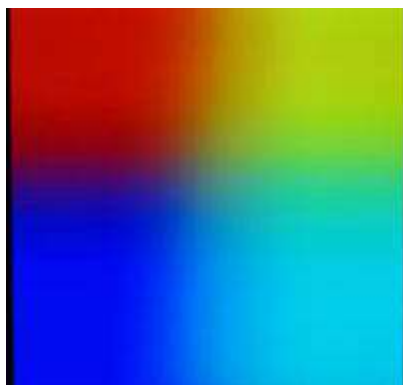


Рисунок 4.1 – Екран завантаження

Крок 6 – запускаємо установщик NOOBS. Після завантаження, програма просить обрати потрібну операційну систему. Вибираємо Raspbian, як показано на рисунку 4.2.

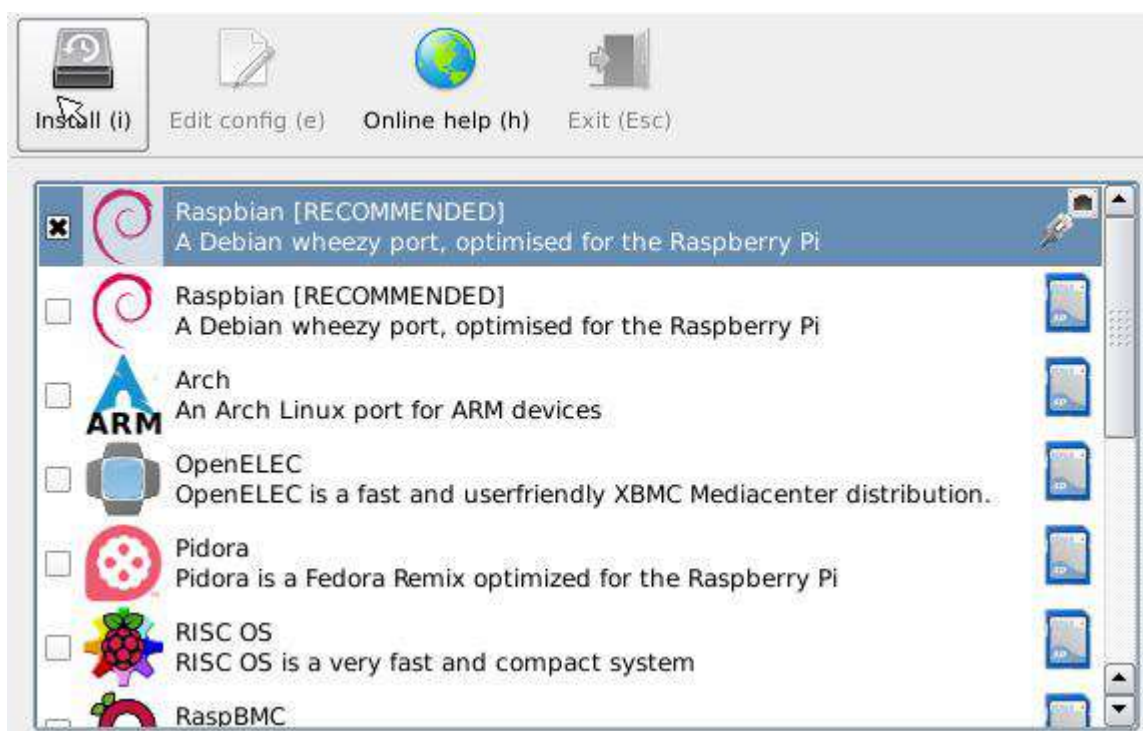


Рисунок 4.2 – Установник NOOBS

Підтверджуємо установку ОС Raspbian натисканням кнопки «Yes».

Крок 6 – копіюємо файли операційної системи (рис. 4.3). Це займає хвилин 15-20.

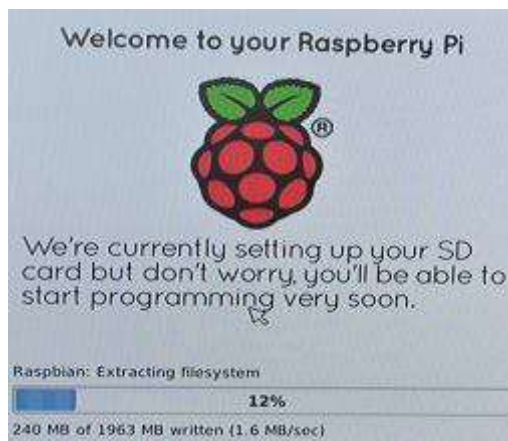


Рисунок 4.3 – Копіювання файлів ОС Raspbian

Крок 7 – по завершенню копіювання й установки на екрані буде виведене вікно, представлене на рисунку 4.4. Натисніть «ОК».



Рисунок 4.4 – Вікно успішної установки ОС Raspbian.

Після виконання всіх кроків, на Raspberry Pi успішно встановлений ОС Raspbian і можна працювати з робочим столом, інтерфейс якого представлено на рисунку 4.5.

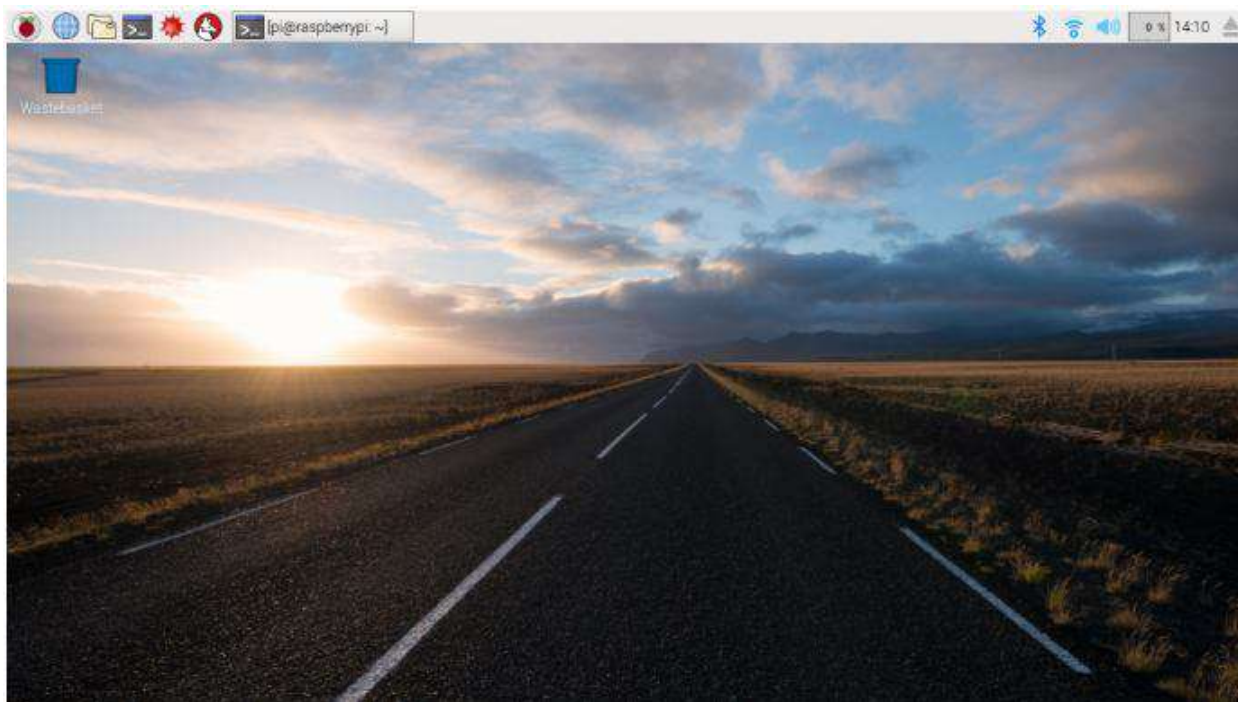


Рисунок 4.5 – Інтерфейс робочого стола ОС Raspbian

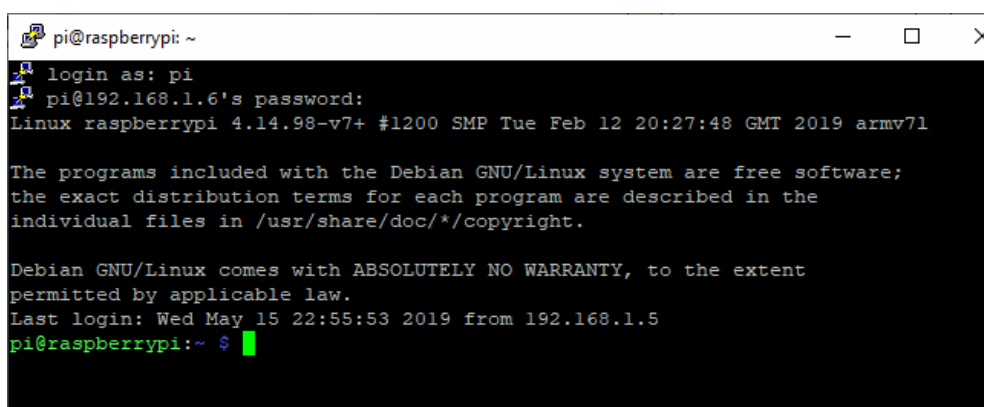
По завершенню процедури установки ОС Raspbian, Raspberry Pi повністю готовий до роботи. У системі вже встановлені різні програмні додатки й середовища розробок, серед них:

- мова програмування Python версій 2 і 3 [272];
- візуальна мова програмування Scratch [273];
- засоби розробки Java додатків: Bluej Java IDE, Greenfoot Java IDE [274];
- засіб розробки Geany Programmer's Editor [275];
- пакет Mathematica [276];
- пакет офісних додатків LibreOffice [277];
- VNCViewer [278];
- браузер Chromium;
- поштовий клієнт Claws Mail;
- гра Minecraft Pi.

4.2 Базові команди Linux для Raspberry Pi

При настроюванні й роботі з Raspberry Pi по необхідності прийдеться часто користуватися командним рядком або *Terminal*. Більшість початківців користувачів Raspberry Pi намагаються мінімізувати використання *Terminal* (рис. 4.6), але це є помилкою [279].

Для того, щоб користуватися терміналом, досить знати кілька базових команд. Після того, як встановлюється підключення по протоколу SSH [280], ці команди дозволять прямо з комп'ютера встановлювати на Pi нове ПЗ, створювати файли / папки та запускати будь-які скрипти без використання графічної оболонки Raspbian OS.



```

pi@raspberrypi: ~
login as: pi
pi@192.168.1.6's password:
Linux raspberrypi 4.14.98-v7+ #1200 SMP Tue Feb 12 20:27:48 GMT 2019 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed May 15 22:55:53 2019 from 192.168.1.5
pi@raspberrypi:~ $

```

Рисунок 4.6 – Terminal на базі вилученого з'єднання через Putty

На першому кроці розглянемо роботу команди *pwd*. У результаті термінал напише */home/pi*. Прямі слеш завжди використовуються для позначення папок і файлів, що перебувають усередині папок. У цьому випадку поточною робочою директорією є *pi*, яка перебуває в директорії *home*, яка перебуває в корені файлової системи.

Слово *pi*, що розміщується перед *@* – це ім'я користувача, під яким він зайшов у систему.

Команди Linux чутливі до регістру, і це значить, що **PWD**, **Pwd**, **pwd** та інші варіації не будуть ідентичні **pwd**. Те ж саме стосується й усіх інших

команд Linux [281].

Найбільш часто використовувані команди:

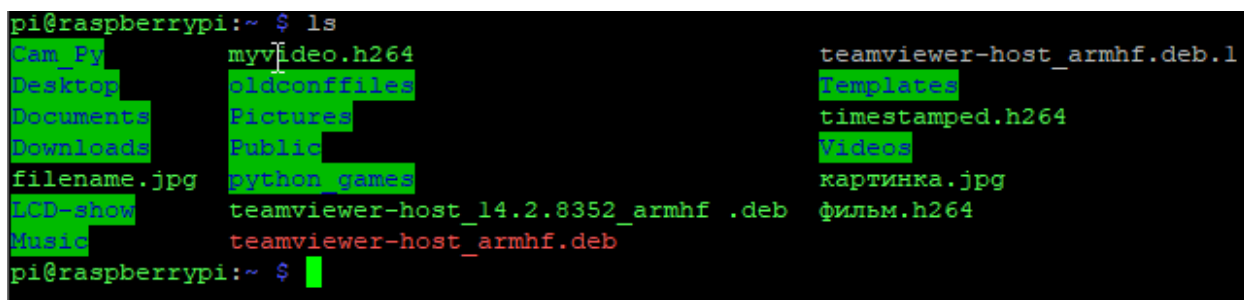
ls – використовується, щоб показати вміст директорії;

cd – дозволяє переміститися з однієї директорії в іншу.

Завантаживши термінал в перший раз, користувач опиниться в домашній директорії (/home/pi). Отже, щоб побачити список файлів і папок, що перебувають у робочій директорії, необхідно в термінал задати команду *ls*:

```
pi@raspberrypi:~ $ ls
```

Результат виконання даної команди представлено на рисунку 4.7



```
pi@raspberrypi:~ $ ls
Cam_Py      myvideo.h264      teamviewer-host_armhf.deb.1
Desktop     oldconffiles      templates
Documents   Pictures           timestamped.h264
Downloads   Public            Videos
filename.jpg python_games       картинка.jpg
LCD-show    teamviewer-host_14.2.8352_armhf .deb  фильм.h264
Music       teamviewer-host_armhf.deb
pi@raspberrypi:~ $
```

Рисунок 4.7 – Результат виконання команди *ls* у директорії /home/pi

Для створення нової директорії необхідно ввести команду *mkdir*, синтаксис якої наведено нижче:

```
mkdir NewFolder
```

NewFolder – ім'я директорії

Для навігації по файловій системі використовується команда *cd*, після якої через пробіл вказується назва директорії, у яку необхідно переміститися.

```
pi@raspberrypi:~ $ cd NewFolder
```

```
pi@raspberrypi:~/NewFolder $
```

Ця команда дає можливість переміщення в папку *NewFolder*.

При роботі з терміналом можна користуватися одним хитрим прийомом, що працює за принципом автодоповнення тексту. Для цього

використовується клавіша “Tab” яка підсвічує назви директорій і файлів за початковими символами.

Для створення нового файлу використовується команда *touch*, синтаксис написання якої представлений нижче:

```
pi@raspberrypi:~/Newfolder $ touch Newfile.txt
```

Окремі файли можна копіювати за допомогою команди *cp*, після якої через пробіл пишеться ім'я файлу. Крім того, цією командою можна користуватися для перейменування файлів:

```
pi@raspberrypi:~/Newfolder $ cp Newfile.txt Otherfile.txt
```

Видалення файлу здійснюється за допомогою команди *rm*, після якої через пробіл пишеться ім'я файлу, що видаляється:

```
pi@raspberrypi:~/Newfolder $ rm Newfile.txt
```

Переміщення файлів здійснюється за допомогою команди *mv*:

```
pi@raspberrypi:~/Newfolder $ mv Otherfile.txt /home/pi
```

За замовчуванням в OS Raspbian і багато інші дистрибутивів Linux вбудований простий текстовий редактор з назвою *nano*. Для відкриття файлу в редакторі *nano* використовується наступний синтаксис:

```
pi@raspberrypi:~ $ nano namefile.txt
```

У результаті *nano* перейде по зазначеному шляху та відкриє файл, якщо той існує. Якщо його не існує, *nano* запустить у цій директорії новий буфер із цією назвою файлу. Приклад виконання команди представленої вище показано на рисунку 4.8.

Зверху інтерфейсу редактора *nano* написана версія програми, назва та розширення файлу, що редагується, а також інформація про те чи був цей файл модифікований після останнього збереження. При роботі з новим файлом, який ще не був збережений, замість назви файлу побачите напис «New Buffer». Далі можна побачити вміст даного файлу.

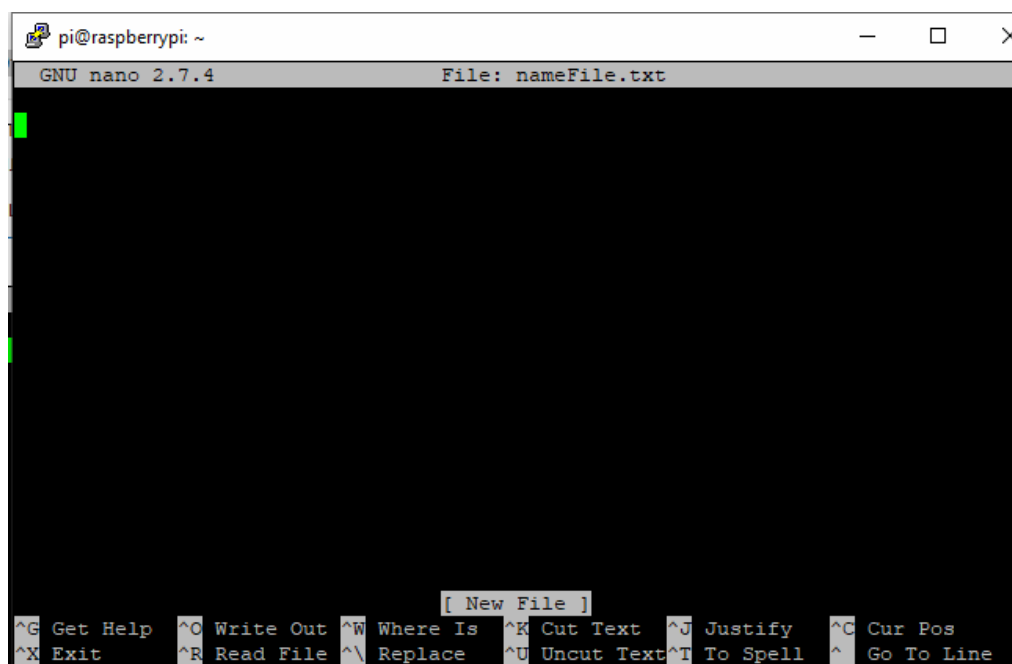


Рисунок 4.8 – Приклад виконання команди *nano namefile.txt*

Редактор *nano* підтримує безліч функцій зі збереження, виходу, пошуку і т.п., тому для користувача в двох самих нижніх рядках містяться підказки для деяких гарячих клавіш, які наведено на рисунку 4.9.

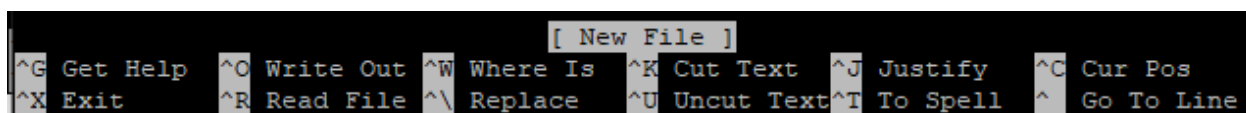


Рисунок 4.9 – Підказка гарячих клавіш редактору *nano*

Символ «^» означає клавішу *Ctrl*.

Клавіша *Shift* для гарячих клавіш не використовується. Крім того, усі гарячі клавіші використовують нижній регістр і незмінені цифрові клавіші, так що *Ctrl+G* – це не *Ctrl+Shift+G*. Для виклику підказки в редакторі *nano* натисніть комбінацію клавіш *Ctrl+G*.

Завантаження та установка нового програмного забезпечення на Raspberry Pi і його аналогів, що працюють на основі OS Linux, відбувається методом «*пакетів*» – програм, які можна завантажити з інтернету й установити, просто написавши в терміналі невелику команду.

Для завантаження й установки цих пакетів використовується, як правило, пакетний менеджер, який завантажує й установлює не тільки саме ПЗ, але й допоміжні програми, які називаються «*залежностями*».

Дистрибутив OS Raspbian використовує пакетний менеджер *apt*. Щоб управляти установкою / видаленням ПЗ, користувачеві необхідно авторизуватися як адміністратор (його також називають «суперкористувачем»). Для цього на початку команди потрібно вписати *sudo* (це скорочення від «superuser do»).

Щоб установити ПЗ, необхідно використовувати наступний синтаксис:

```
sudo apt-get install <назва ПЗ>
```

При необхідності запуску ПЗ прямо з термінала, використовується синтаксис *nameprogramm* – *назва ПЗ*:

```
pi@raspberrypi:~ $ nameprogramm
```

Щоб видалити ПЗ з Raspberry Pi, використовується пакетний менеджер *apt*. Синтаксис написання даної команди:

```
sudo apt-get remove <назва ПЗ>
```

Ця команда, втім, не видаляє файли, пов'язані з вилученням ПЗ – логів і конфігураційних файлів. При відсутності необхідності використовувати їх надалі, можна скористатися командою *purge*, яка видаляє й саме ПЗ і допоміжні файли:

```
sudo apt-get purge <название ПО>
```

Не рекомендується видаляти пакети, які не встановлювалися користувачем (адже потрібно добре розуміти, що саме видаляється). Це можуть бути важливі пакети, що йдуть разом з операційною системою Linux, і їх видалення може привести до падіння системи.

Щоб поміняти налаштування Raspberry Pi, можна скористатися інструментом, написаним Алексом Бредбери (Alex Bradbury) [282]. Щоб відкрити цей інструмент, просто впишіть у термінал наступну команду:

```
sudo raspi-config
```

Команда *sudo* необхідна для дозволу авторизації, так як можлива ситуація редагувати файли, якими користувач *pi* не володіє. Після введення команди повинен з'явитися синій екран із сірим вікном, у центрі якого буде перебувати кілька опцій (рис. 4.10).

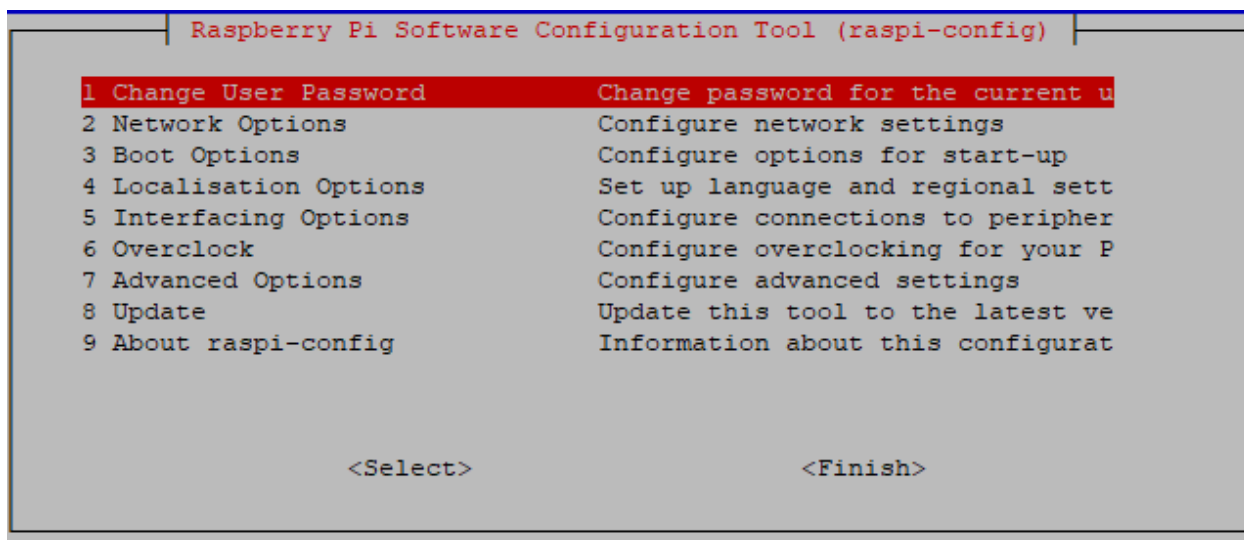


Рисунок 4.10 – Вікно Raspberry Pi Software Configuration Tool

raspi-config – використовується для зміни найпоширеніших налаштувань. Деяким із цих опцій потрібно перезавантаження, щоб зміни набули чинності. Якщо ви поміняли хоча б одне налаштування, то після натискання на кнопку **<Finish>** *raspi-config* запропонує перезавантаження Raspbian OS. Основний інструмент *raspi-config* дозволяє міняти наступні налаштування [283]:

- Change User Password – зміни пароля користувача;
- Boot Options – вибір інтерфейсу, який увімкнеться при завантаженні обладнання;
- Internationalisation Options – тут настраюються годинний пояс, розкладка клавіатури, мова й регіональне налаштування;
- Overclock – зміна тактової частоти Raspberry Pi;
- Advanced Options – додаткові опції;
- About raspi-config – інформація про інструмент raspi-config.

Перезавантаження Raspberry Pi виконується командою:

```
sudo reboot
```

Завершення роботи Raspberry Pi:

```
sudo poweroff
```

Зверніть увагу, фізичне відключення живлення (від джерела) рівнозначно вимкнення комп'ютера натисканням кнопки Power або навіть відключенням від розетки, що може привести до ушкодження файлів.

4.3 OS Raspbian

4.3.1 Установка екранної клавіатури

При використанні Raspberry PI разом із сенсорним дисплеєм зручніше з екранною клавіатурою. Для її установки необхідне виконання наступної послідовності етапів.

На першому етапі оновимо OS Raspbian через Terminal наступними командами:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

Потім за допомогою команди *sudo apt-get install matchbox-keyboard* – проведемо установку пакета екранної клавіатури.

Однією з головних умов, необхідних для відображення в *Menu* екранної клавіатури, буде підтвердження її статусу *Visible*. Для цього необхідно викликати *Main Menu Editor* через *Menu* (рис. 4.11).

Потім обираємо виберіть пункт меню *Accessories* і активуємо «галочку» напроти *Keyboard* (рис. 4.12).

Після цього віртуальна клавіатура буде доступна всім користувачам на всіх вилучених робочих столах за наступним шляхом: *Menu -> Accessories -> Keyboard*.

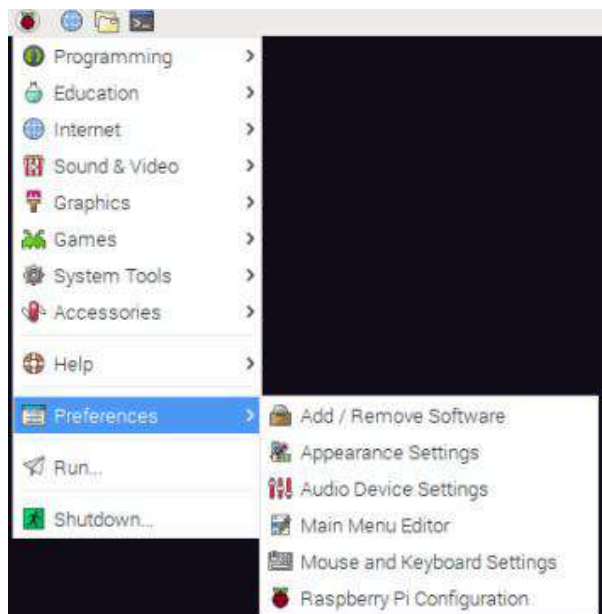


Рисунок 4.11 – Виклик *Main Menu Editor* через *Menu*



Рисунок 4.12 – Вікно *Main Menu Editor*

У підменю *Accessories* можна відразу налаштувати відображення потрібних програм для роботи з OS Raspbian. Це актуально коли використовуються сенсорні екрани з невеликою роздільною здатністю.

4.4 Налаштування віддаленого доступу

4.4.1 Налаштування віддаленого доступу через VNC

Метод, представлений у даному підрозділі, дозволить здійснювати обмежений перелік дій, що включає роботу в терміналі. Він не дозволить запускати повноекранні графічні додатки, такі як ігри або медіаплеєри. Проте, він забезпечить повноцінне оточення, таке ж, як якби воно було виведено прямо на монітор.

При розробці систем керування мобільними роботами, виникне необхідність управляти Raspberry Pi віддалено, будь це транслявання відеопотоку з web-камери для спостереження за навколишнім простором, програмування елементів обробки й прийняття рішень на Python, або налаштування OS Raspbian. Отже на даний момент часу не існує більш зручного методу чим використання віддаленого робочого столу. Наведена послідовність дозволить налаштувати зв'язування Raspberri Pi 3 + ПК на Windows, що перебувають в одній локальній мережі [284].

На першому етапі оновимо OS Raspbian:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

Опція *update* утиліти *apt-get* запускає перевірку в репозиторіях нових версій програм, доступних для завантаження й оновляє кеш у системі. Опцією *upgrade* запускається відновлення системи до останньої версії. Як правило обидві команди використовуються разом. Можна використовувати обидві команди в одному рядку, для послідовного виконання обох команд, для цього потрібно розділити їх крапкою з комою – *sudo apt-get update; sudo apt-get upgrade* або амперсандом – *sudo apt-get update & sudo apt-get upgrade*. Це досить довга процедура, якщо це перше відновлення після установки OS Raspbian.

Для коректного виконання важливих команд використовується комбінація `&&` – наступна команда буде виконана, тільки якщо попередня завершена без помилок: `sudo apt-get update && sudo apt-get dist-upgrade`.

Перед маніпуляціями із сервером необхідно налаштувати статичну IP-адресу обладнання, для того щоб VNC-клієнт міг звертатися до обладнання за конкретною адресою. Наступними діями буде скасування режиму DHCP (англ. Dynamic Host Configuration Protocol) – протоколу динамічного налаштування вузла. Він використовується для автоматичного динамічного присвоєння IP-адреси пристроям мережі. Далі всі параметри мережі встановлюємо вручну [285].

Можна використовувати автоматичний режим DHCP, якщо використовується локальна домашня або лабораторна мережа, але при цьому варто врахувати, що в будь-який момент IP-адреса може бути змінена.

Для цього спочатку потрібно виконати команду (відкрити файл `/etc/dhcpd.conf` для редагування в текстовому редакторі `nano`):

```
sudo nano /etc/dhcpd.conf
```

І в самому кінці файлу під рядком *nohook lookup-hostname* вказати дані своєї мережі й бажана IP-адреса обладнання (вона повинена бути вільною, щоб не було конфліктів, визначити вільну IP-адресу можна за допомогою програми Advanced IP Scanner (<http://www.advanced-ip-scanner.com>)) у мережі:

```
interface eth0
```

```
static ip_address=192.168.1.6
```

```
static routers=192.168.1.1
```

```
static domain_name_servers=192.168.1.1 192.168.1.2
```

Зверніть увагу, що налаштування для Wi-Fi і Ethernet відбувається окремо. Для Wi-Fi у першому рядку коду потрібно вказати *interface wlan0*.

Після того як усе налаштовано, потрібно перезавантажити систему й переконатися, що налаштовані параметри набули чинності (рис. 4.13).

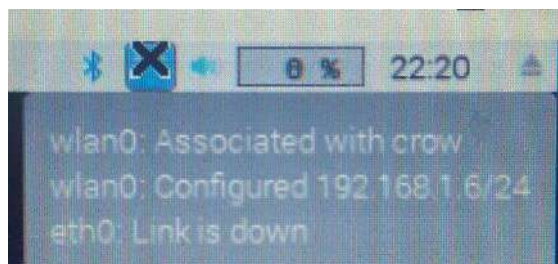


Рисунок 4.13 – Прийняті зміни у файлі *dhcpcd.conf* при підключенні *wlan0*.

Установлену IP-адресу можна завжди довідатися командою:

hostname -I

Наступним етапом необхідно налаштувати **VNC Server**. Починаючи з версії 4.8 (Raspbian Jessie With Desktop) VNC Server установлений в системі. Якщо його з якихось причин немає, його можна встановити командою:

sudo apt-get install realvnc-vnc-server

Після цього його потрібно включити в налаштуваннях Start → Preferences → Raspberry Pi Configuration, на вкладці Interfaces, установити галочку Enabled напроти VNC (рисунок 4.14).

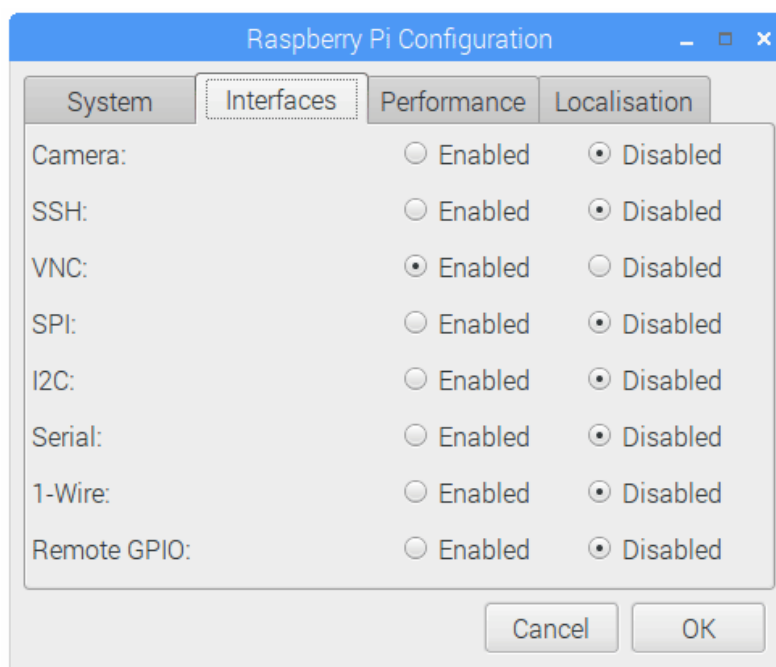


Рисунок 4.14 – Вікно Raspberry Pi Configuration

Якщо для налаштування VNC використовується термінал через протокол SSH на базі Putty Вікно Parberry Pi Configuration можна викликати командою: *sudo raspi-config* ->(меню Interfacing Options)

VNC-Сервер буде запущений автоматично, якщо цього не відбулося, його можна запустити вручну:

```
vncserver
```

Перед запуском сервера необхідно налаштувати зручний для роботи дозвіл віддаленого робочого столу. Для цього необхідно внести зміни у файл */boot/config.txt* за допомогою редактора *nano*:

```
sudo nano /boot/config.txt
```

Необхідно додати рядок:

```
hdmi_force_hotplug=1
```

```
hdmi_ignore_edid=0xa5000080
```

```
hdmi_group=2
```

```
hdmi_mode=82
```

У цьому випадку *hdmi_mode=82* - 1920×1080, 60Hz, 1080p, *hdmi_force_hotplug* = 1. Приклад наведено на рисунку 4.15.

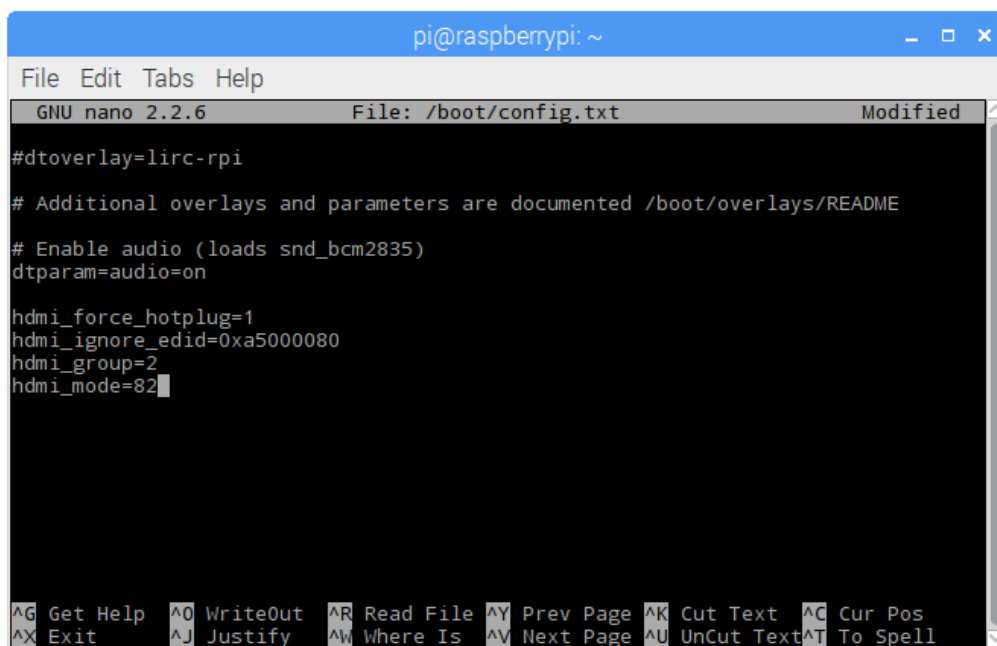


Рисунок 4.15 – Налаштування дозволу віддаленого робочого столу

Після цього зберігаємо внесені зміни *config.txt* за допомогою комбінації клавіш **Ctrl+X, Y(yes)**.

Raspberry Pi за замовчуванням вважає, що до HDMI-роз'єму підключений монітор. Якщо монітор підключається по RCA-роз'єму, ці рядки необхідно вилучити.

Альтернативно можна задати дозвіл віддаленому столу за допомогою команди:

```
tightvncserver :1 -geometry 1360x768 -depth 24
```

Як можна бачити дана команда створює робочий стіл під номером 1 з дозволом 1360x768 на 24 години, якщо протягом даного часу не буде приєднань налаштування скидається.

При першому створенні віддаленого робочого стола в терміналі необхідно ввести login: **** і password:***** (не менше 8 символів) [284].

На наступному етапі перейдемо до налаштування **VNC Viewer** на клієнтській стороні OS Windows (на прикладі Windows 10), але насправді вона може бути будь-яка, у тому числі й Android. Для початку необхідно завантажити VNC Viewer (<https://www.realvnc.com/en/connect/download/viewer/windows/>) і встановити. Першим кроком необхідно створити нове підключення, як показано на рисунку 4.16.

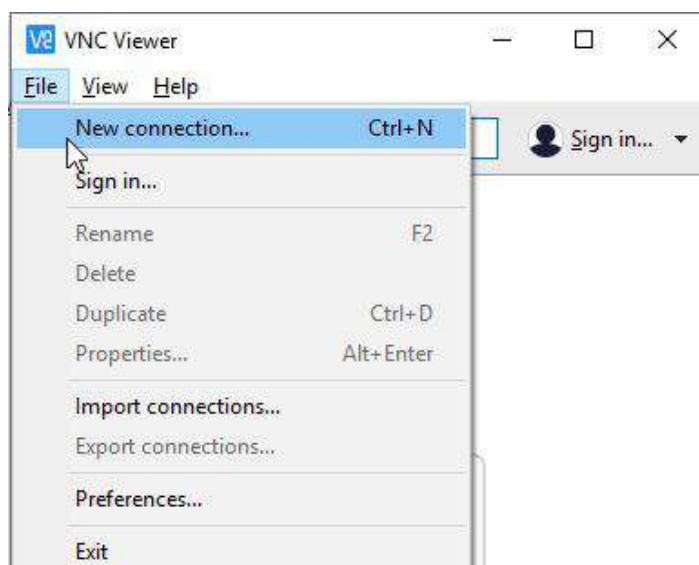


Рисунок 4.16 – Створення нового підключення

У налаштуваннях з'єднання потрібно вказати тільки IP-адресу обладнання, до якого буде здійснене підключення — вона задається у файлі *dhcpcd.conf* або упізнається динамічно через програму *Advanced IP Scanner*. Через двокрапку вказуємо номер дисплея, заданий при запуску VNC-сервера (за замовчуванням 0 — можна не вказувати). У полі *name* можна вказати ім'я з'єднання, наприклад, Raspberry Pi 3. Якщо планується працювати тільки з одним робочим столом, то можна вказувати IP-адресу (у полі VNC Server) без його номера (192.168.1.6). Але якщо робочих місць буде декілька (кожна нова команда *vncserver* створює нове місце), підключатися до кожного з них потрібно буде із вказівкою його номера після IP-адреси через двокрапку: 192.168.1.6:1, 192.168.1.6:2 і т.д. (рис. 4.17).

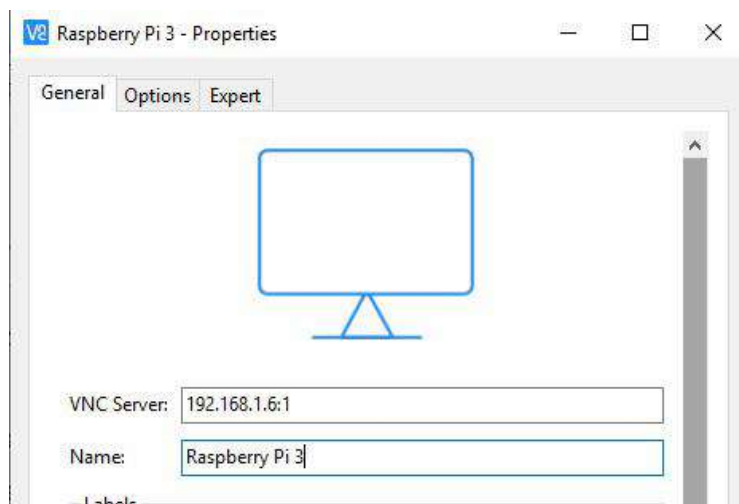


Рисунок 4.17 – Налаштування VNC

Знову створене з'єднання з'являється в головному вікні VNC Viewer (рис. 4.18).

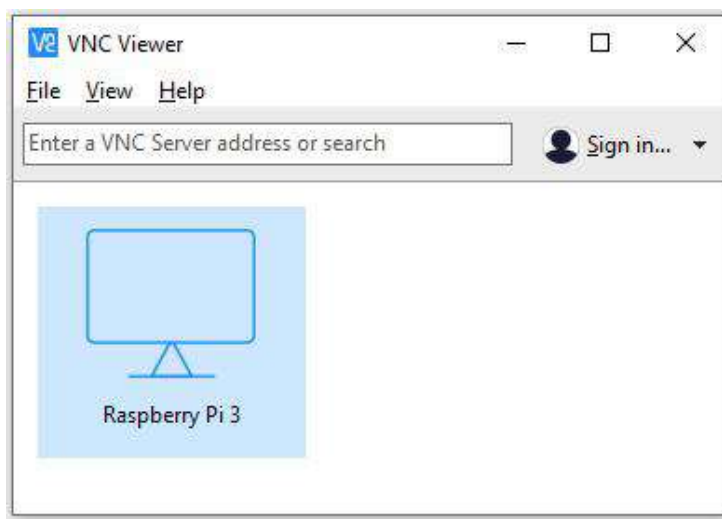


Рисунок 4.18 – Головне вікно VNC Viewer

Якщо всі налаштування зроблені коректно, то при подвійному кліку повинне з'явитися вікно із запитом імені користувача й пароля (рис. 4.19). Завжди, за замовчуванням: логін – pi, пароль – raspberry.

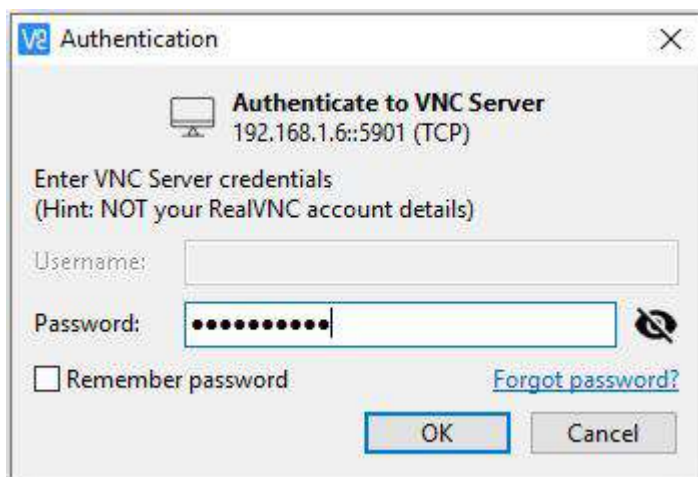


Рисунок 4.19 – Вікно авторизації

Можна встановити галочку *Remember Password* – у цьому випадку вводити пароль більше не буде потрібно.

Після введення пароля, з'являється віддалений робочий стіл системи (рис 4.20) із заданим дозволом.

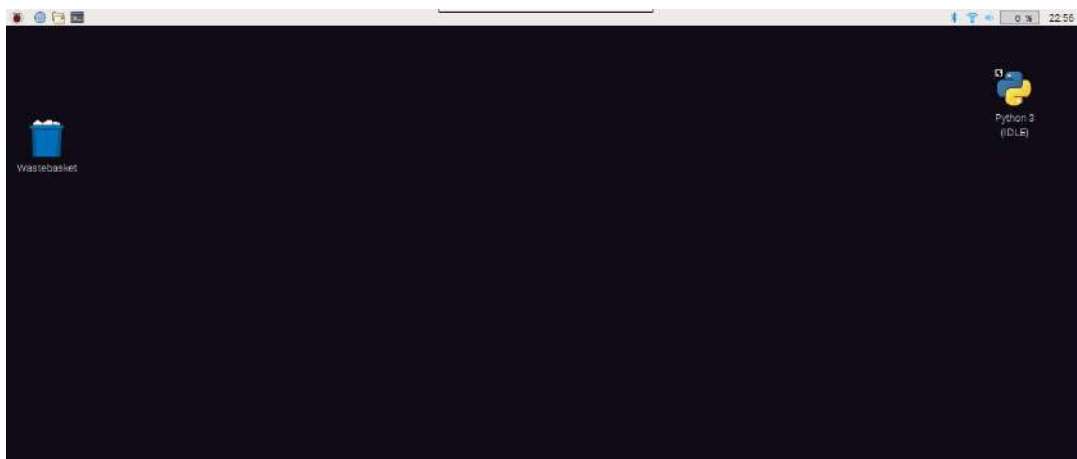


Рисунок 4.20 – Створений віддалений робочий стіл

Віддалений робочий стіл зберігається доти, поки його явно не буде знищено. Запустіть наступну команду, якщо впевнені, що він більше не потрібний:

```
vncserver -kill :<display-number>
```

Для зупинки VNC сервера потрібно виконати команду:

```
sudo systemctl stop vncserver-x11-serviced.service
```

Докладніше про команди можна подивитися на офіційній сторінці <https://www.realvnc.com/en/docs/raspberry-pi.html>.

4.4.2 Віддалений робочий стол через Internet

Іноді виникає необхідність дистанційного доступу до RPi через домашню мережу. Існує 2 методи:

- через виділену статичну IP-адресу [286];
- за допомогою програми TeamViewer [287].

Перший спосіб складний і тут не буде розглянутий. Другий спосіб дуже простий і ефективний. Для його реалізації необхідно завантажити TeamViewer Host для Debian (armv7 32bit). Далі встановити додаток за допомогою утиліти прямої установки додатків *dpkg* з файлу *.deb*, попередньо ознайомившись з інструкціями:

```
wget https://download.teamviewer.com/download/linux/teamviewer-  
host_armhf.deb
```

```
sudo dpkg --install teamviewer-host_armhf.deb
```

Але оскільки недоліком використання утиліти *dpkg* є те, що вона працює з окремими пакетами й не обробляє залежності автоматично, установка припиниться. При цьому утиліта надасть список залежностей, які можна встановити командою:

```
sudo apt-get install -f
```

Для налаштування роботи Teamviewer скористаємося вже налаштованим VNC Viewer. На рисунку 4.21 представлений фрагмент віртуального робочого стола із шляхом виклику TeamViewer 14 Host.

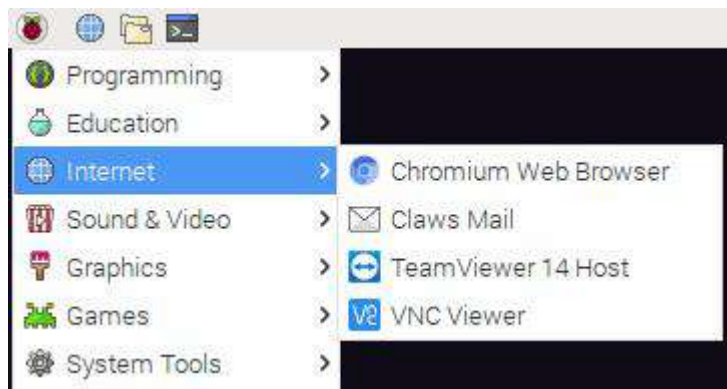


Рисунок 4.21 – Фрагмент віртуального робочого стола із шляхом виклику Teamviewer 14 Host

3.5 Налаштування Samba на OS Raspbian

При розробці проектів, пов'язаних з мобільними роботами, часто виникає необхідність взаємодії з файловою системою Raspbian OS через робочий комп'ютер на базі OS Windows. Для того щоб реалізувати Raspberry Pi 3 у файловий сервер, з дистанційним доступом до файлів по локальній мережі, знадобиться встановити файловий сервер Samba [288-291]. Він дозволяє здійснювати кроссплатформну файлову взаємодію в мережі й працює по протоколу SMB/CIFS [292].

На першому етапі зробимо відновлення системи за допомогою команд:

```
sudo apt-get update
```

sudo apt-get dist-upgrade

Далі, після відновлення OS Raspbian, проводимо установку Samba. Samba – пакет програм, які дозволяють звертатися до мережних дисків і принтерів на різних операційних системах по протоколу SMB/CIFS. Має клієнтську й серверну частини. Є вільним програмним забезпеченням, випущеним під ліцензією GPL. Samba працює на більшості Unix-подібних систем, таких як Linux, Posix-Сумісних Solaris і Mac OS X Server, на різних варіантах BSD; в OS/2 вмонтовано Samba-клієнт, що є плагіном до віртуальної файлової системи Netdrive [293, 294]. Samba включена практично в усі дистрибутиви Linux. Установку Samba (рис. 4.22) проводимо через термінал SSH або віртуальний робочий стіл за допомогою команди:

sudo apt-get install samba samba-common-bin



Рисунок 4.22 – Команда для установки Samba у терміналі

Після установки Samba необхідно створити папку на жорсткому диску Raspberry Pi (Microsd з OS), доступ до якої необхідно одержати. У нашій випадку це буде папка *share* у кореневій директорії:

sudo mkdir -m 1777 /share

Ця команда встановлює *Sticky bit* (1), який використовується в основному для каталогів, щоб захистити в них файли [295]. З такого каталогу користувач може вилучити тільки ті файли, власником яких він є. Прикладом може служити каталог */tmp*, у якому запис відкритий для всіх користувачів, але небажане видалення чужих файлів. Використовується щоб запобігти випадковому видаленню каталогу й дає кожному користувачеві права на читання / запис / виконання (777).

Тепер Samba потрібно налаштувати. Для цього відкриваємо файл конфігурації:

sudo leafpad/etc/samba/smb.conf – через текстовий графічний редактор по віддаленому робочому столу або напрямую (рис. 4.23).

sudo nano/etc/samba/smb.conf – через термінальний текстовий редактор.

Зверніть увагу на необхідність створення резервної копії файлу конфігурації Samba за.

Додаємо в кінець файлу наступні 10 рядків налаштування:

[share]

Comment = Pi shared folder

Path = /share

Browseable = yes

Writeable = yes

only guest = no

create mask = 0777

directory mask = 0777

Public = yes

Guest ok = yes

У такий спосіб дозволяємо всім користувачам, у тому числі й гостям вашої локальної мережі, мати повний доступ до читання, запису та виконання файлів у загальному каталозі */share* (рис. 4.23).



```

*smb.conf
File Edit Search Options Help
path = /var/lib/samba/printers
browseable = yes
read only = yes
guest ok = no
# Uncomment to allow remote administration of Windows print drivers.
# You may need to replace 'lpadmin' with the name of the group your
# admin users are members of.
# Please note that you also need to set appropriate Unix permissions
# to the drivers directory for these users to have write rights in it
; write list = root, @lpadmin

[share]
Comment = Pi shared folder
Path = /share
Browseable = yes
Writeable = Yes
only guest = no
create mask = 0777
directory mask = 0777
Public = yes
Guest ok = yes

```

Рисунок 4.23 – Налаштування Samba сервера до доступу до папки *Share* для всіх користувачів

Що б внесені зміни набули чинності необхідно перезавантажити Samba:

```
sudo /etc/init.d/samba restart
```

4.5.1 Налаштування доступу до існуючих папок OS Raspbian

У більшості випадків при роботі із проектами в команді може перебувати велика кількість людей для яких необхідно відкрити доступ до вже існуючих папок, наприклад */home/pi* [296]. Для цього необхідно внести зміни у файл конфігурації *smb.conf*. Для цього відкриваємо його за допомогою:

sudo leafpad/etc/samba/smb.conf – через текстовий графічний редактор на віддаленому робочому столі або на пряму (рис. 4.23);

sudo nano/etc/samba/smb.conf – через термінальний текстовий редактор та додаємо в кінець файлу наступні налаштування, при чому параметру Path привласнюємо шлях до папки, яку прагнемо зробити доступною – */home/pi*:

```
[share]
```

Comment = Pi shared folder

Path = /home/pi

Browseable = yes

Writeable = yes

only guest = no

create mask = 0777

directory mask = 0777

Public = yes

Guest ok = yes

Потім перезавантажуємо Samba за допомогою команди:

sudo /etc/init.d/samba restart

Папка стане доступною всім користувачам мережі, але тільки для читання. Це пов'язане з тим, що всі папки й файли були створені раніше й для того, щоб відкрити повний доступ до них, потрібно перепризначити права на всі елементи розширеної папки:

sudo chmod -R 777 /home/pi

Тут необхідно мати на увазі, що файли, створені безпосередньо в Raspbian OS (наприклад, скетч Python) після відкриття доступу, не будуть мати дозволу на зміни з боку зовнішніх комп'ютерів у мережі. Для того, щоб призначити права на зміну цих файлів, потрібно знову повторити процедуру зміни прав: *sudo chmod -R 777 /home/pi*.

Якщо необхідно надати доступ до кореневої папки (або до всіх папок), то у файлі налаштувань необхідно змінити рядок *Path = /any_path* на *Path = /*. При цьому прийдеться повторити процедуру зміни прав на папку (*атрибут - R* вказує, що команду слід застосувати також до всіх вкладених елементів).

Для обмеження доступу до відкритих ресурсів Raspberry Pi3 необхідно, по-перше, заборонити доступ гостям. Для цього з файлу конфігурації потрібно вилучити рядок *Guest ok = yes*. По-друге, потрібно задати пароль Samba для доступу до папки *pi* за допомогою команди:

sudo smbpasswd -a pi

Для застосування змін необхідно перезавантажити сервер Samba за допомогою команди:

```
sudo /etc/init.d/samba restart
```

У випадку зміни прав на всі папки: *sudo chmod -R 777 /* автоматично будуть переписані права й root-користувача (папка *usr/bin/sudo*). Після цієї процедури виконання будь-якої команди під суперюзером *sudo* буде неможливо. Будь-яка спроба використання команди *sudo* буде завершена з відображенням помилки:

```
sudo: /usr/bin/sudo повинен належати користувачеві з uid 0 і мати біт setuid
```

Тому рекомендується вибірково обирати папки для зміни прав:

```
sudo chmod -R 777 /one_folder
```

4.6 Створення резервної копії OS Raspbian

Створення резервної копії (“бекапа”) з Raspberry Pi представляє із себе образ, знятий з microSD-картки із встановленою й сконфігурованою OS Raspbian і набором додатків. Отже користувач створює власну версію дистрибутива Raspbian або робить унікальне складання з оригінального OS Raspbian із встановленим у цьому випадку набору сторонніх додатків, необхідних для розв'язків певних завдань, поставлених перед користувачем. Для створення резервної копії необхідні наступні елементи:

- Raspberry Pi 3 Model B;
- картридер або здатне його замінити обладнання;
- комп'ютер під керуванням Windows, Linux або macOS.

Розглянемо послідовність дій для створення резервної копії під OS Windows. Для цього буде потрібна безкоштовна програма Win32 Disk Imager, скачати яку можна сайті Sourceforge: <https://sourceforge.net/projects/win32diskimager/> (рис. 4.24) [297].

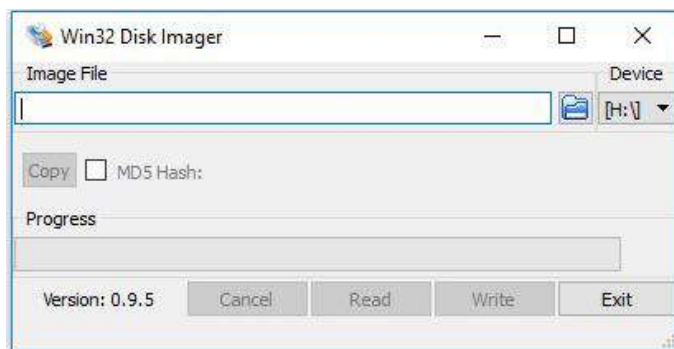


Рисунок 4.24 – Загальний вид програми Win32 Disk Imager

При встановленні у картридер microSD-картки з Raspberry Pi у поле *Image File* уводимо назву файлу, у який буде записаний образ, і натискаємо кнопку “Read”. Програма повністю зчитає вміст microSD в Img-файл. Тому невигідно використовувати в якості системного накопичувача Raspberry Pi карти пам'яті великих розмірів – вони всі будуть зчитуватися цілком, а тримати на комп'ютері диску “бекапи” вагою по 32-64Гб кожний, досить не зручно. Оптимальний розмір microSD під Raspberry Pi, для розв'язку завдань розглянутих в даній книзі, повинне бути у межах 8–16Гб. Більше – уже немає особливої потреби. Для зберігання файлів краще використовувати USB-накопичувачі.

Відновлення Raspberry Pi з бекапа відбувається через ту ж програму Win32 Disk Imager. Потрібно вказати шлях до збереженого файлу образу й натиснути кнопку “Write”.

Після відновлення системи з бекапа злітають створені з'єднання з Bluetooth-обладнаннями – потрібно заново проводити сполучення, але всі інші налаштування зберігаються.

Розглянемо послідовність дій для створення резервної копії під OS Linux. В Linux-системах зробити “бекап” Raspberry Pi можна за допомогою системної утиліти *dd*.

Для цього потрібно ввести в терміналі наступну команду:

```
sudo dd if=/dev/mmcblk0 of=/home/Username/Desktop/raspberrypi-backup.img
```

Де в “*if=*” прописується шлях до вставленої в картридер microsd-карти, а в “*of=*” прописується шлях до файлу образу, у який вміст цієї карти буде вписуватися.

Відповідно, замість */dev/mccblk0* і */home/Username/Desktop/ raspberrypi-backup.img* потрібно ввести свій шлях.

Шлях до змонтованої microsd-карті можна подивитися виконанням консольної команди:

```
ls -l /dev/disk/by-uuid
```

Відновлення Raspberry Pi з бекапа проводиться за допомогою тієї ж утиліти *dd*, тільки міняються місцями значення “*of*” і “*if*”:

```
sudo dd if=/home/Username/Desktop/raspberrypi-backup.img of=/dev/mccblk0.
```

Розглянемо послідовність дій для створення резервної копії *macos*. Оскільки *macos* теж є * nix-системою, процес бекапа Raspberry Pi у цій операційній системі не буде кардинально відрізнятися від вищеописаної процедури “бекапа” під Linux.

Тільки шлях до змонтованої microsd-карті можна переглянути через програму Disk Utility (дискова утиліта), інтерфейс якої представлено на рисунку 4.25 [298].



Рисунок 4.25 – Загальний вид програми Disk Utility для macOS

Потім виконується команда *dd* для запису образу карти пам'яті у файл:

```
dd if=/dev/disk2 of=raspberrypi-backup.img
```

Для відновлення карти пам'яті зі знятого образу OS Raspbian необхідно ввести наступну команду:

```
dd if=raspberrypi-backup.img of=/dev/disk2.
```

Перший “бекап” рекомендується зробити відразу після завершення установки й налаштування OS Raspbian, і надалі робити бекапи не рідше, чим раз в 2-4 тижні.

4.7 Установка бібліотеки OpenCV

У цьому розділі книги розглянемо особливості установки бібліотеки комп'ютерного зору OpenCV і використання її на Raspberry Pi 3 за допомогою мови Python [299-302]. Це бібліотека з відкритим вихідним кодом, яка дозволяє реалізовувати різні алгоритми обробки відео- зображень, за допомогою стандартних типів даних, функцій і класів.

Варто зауважити, що принцип установки бібліотеки OpenCV для мови Python (OS Raspbery) і принцип установки бібліотеки OpenCV для мови Python (OS Windows famely) неоднакові.

Для зручності опису процесу установки і компіляції ядра, даний матеріал буде представлений у вигляді кроків.

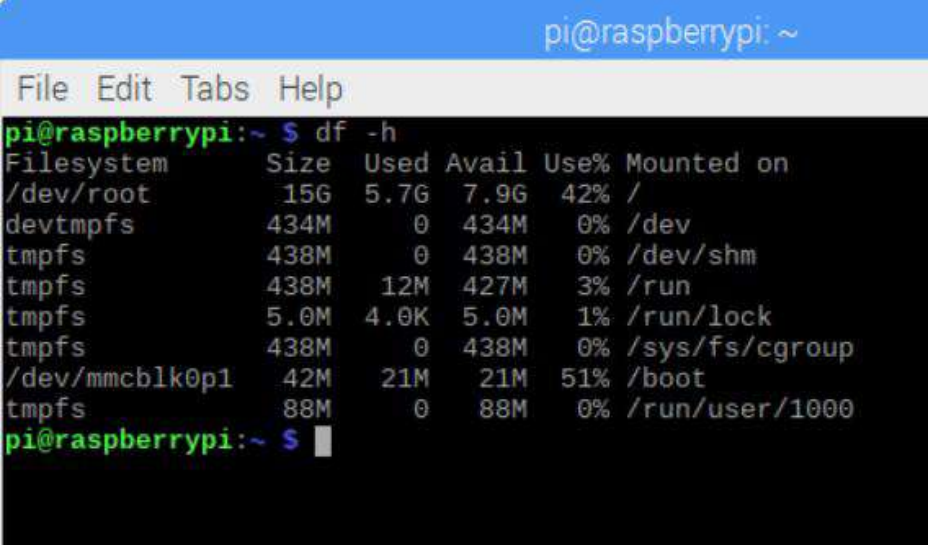
Крок 1. Підготовка перед установкою.

Бібліотека OpenCV для своєї установки вимагає значний обсяг вільного дискового простору – приблизно 6,5–7 Гб (з них близько 6 Гб можна буде видалити після установки). Тому, для використання OpenCV, рекомендується використовувати SD-карту ємністю не менше 16 Гб [303].

Перевірити наявність вільного місця на SD-картці (рисунок 4.26), можна, скориставшись утилітою *df* з параметром *-h*:

```
df -h
```

Якщо з якихось причин, на момент початку установки бібліотеки OpenCV, файлова система не розширена на всю SD-карту, це необхідно зробити як описано вище.



```

pi@raspberrypi:~ $ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root        15G   5.7G   7.9G  42% /
devtmpfs        434M    0   434M   0% /dev
tmpfs           438M    0   438M   0% /dev/shm
tmpfs           438M   12M   427M   3% /run
tmpfs           5.0M   4.0K   5.0M   1% /run/lock
tmpfs           438M    0   438M   0% /sys/fs/cgroup
/dev/mmcblk0p1   42M   21M   21M  51% /boot
tmpfs           88M    0    88M   0% /run/user/1000
pi@raspberrypi:~ $

```

Рисунок 4.26 – Результати аналізу вільного місця

Для того, щоб звільнити додаткове місце на SD-карті, можна видалити непотрібні пакети, наприклад Wolfram Mathematica (близько 700 Мб) або LibreOffice (близько 200 Мб) наступними командами:

```
sudo apt-get purge wolfram-engine
```

```
sudo apt-get purge libreoffice*
```

Те ж саме можна зробити за допомогою вікна Add/Remove Software як показано на рисунку 4.27.

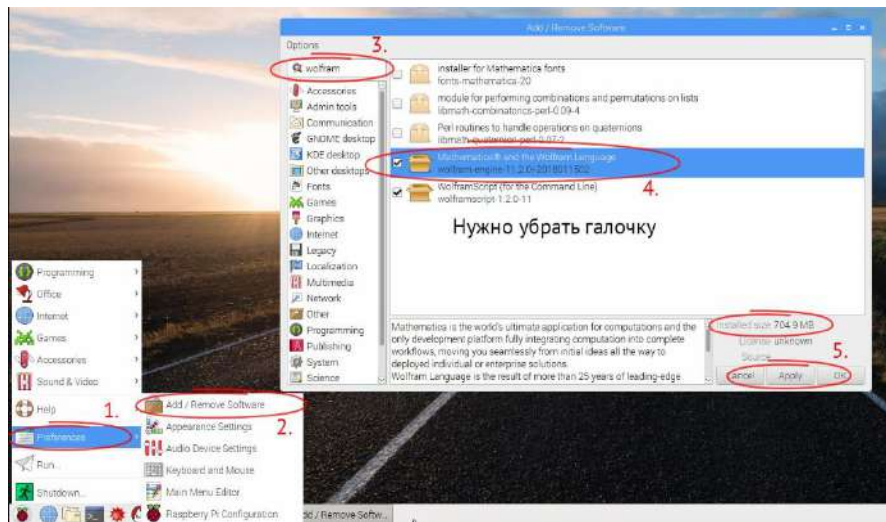


Рисунок 4.27 – Видалення програм через вікно Add/Remove Software

За допомогою команди *clean* звільняється дисковий простір за рахунок видалення отриманих *.deb*-пакунку пакетів в локальному репозиторії (очищення локального кеша):

```
sudo apt-get clean
```

Командою *autoremove* автоматично будуть видалені встановлені залежні від інших, але більше не потрібні, «осиротілі» пакети. Це дозволить вивільнити ще близько 700–800 Мб:

```
sudo apt-get autoremove
```

Перед усіма маніпуляціями, як зазвичай, необхідно оновити систему командами *sudo apt-get update* і *sudo apt-get dist-upgrade*, а також перезавантажити операційну систему.

Крок 2. Встановлення залежних пакетів.

Перед початком установки потрібно встановити необхідні для роботи бібліотеки OpenCV пакети, включаючи додаткові пакети для роботи з зображеннями, відео і т.д [304-306]. Назва додаткових пакетів і їх опис представлено в таблиці 4.1.

Таблиця 4.1 – Опис додаткових пакетів для роботи з бібліотеки OpenCV

Назва пакета	Опис пакета
1	2
Cmake [307]	Утиліта для автоматичного складання програми з вихідного коду
cmake-curses-gui [308]	Пакет GUI (графічний інтерфейс) для cmake
libgtk2.0-dev [309]	GTK+ інструментарій для створення графічних користувацьких інтерфейсів
libjpeg-dev [310]	Бібліотеки для роботи з різними форматами зображень
libjasper-dev [311]	JasPer – набір програм для кодування та обробки зображень
libavcodec-dev [312]	Бібліотека кодеків от Libav (аудіо / відео)
libswscale-dev [313]	Бібліотека для виконання високооптимізації (збільшення або зменшення) зображення
libv4l-dev [314]	Набір бібліотек, для роботи з пристроями video4linux2
libx264-dev [315]	Бібліотека кодування для створення відеопотоків H.264 (MPEG-4 AVC)
libxvidcore-dev [316]	Відеокодек MPEG-4 (Xvid)
Gfortran [317]	Матричні перетворення
libatlas-base-dev [318]	Підпрограми лінійної алгебри

Для роботи з бібліотекою OpenCV мінімально необхідно встановити наступний набір пакетів за допомогою команд:

```
sudo apt-get install cmake cmake-curses-gui libgtk2.0-dev
```

```
sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev libx264-dev libxvidcore-dev
```

```
sudo apt-get install libjpeg-dev libpng12-dev libtiff5-dev libjasper-dev
```

```
sudo apt-get install gfortran libatlas-base-dev
```

Не буде зайвим встановити додаткові пакети для роботи з відео, закодованими різними кодеками:

```
sudo apt-get install -y libdc1394-22-dev libavresample-dev libgphoto2-dev  
ffmpeg libgtk-3-dev
```

Якщо планується 3D-візуалізація або 3D-моделювання (в т.ч. складних процесів), знадобиться пакет VTK (Visualization ToolKit, – інструментарій візуалізації, відкрита кроссплатформенная програма для тривимірного моделювання, обробки зображень і прикладної візуалізації [319-321]:

```
sudo apt-get install -y libvtk6-dev
```

Далі потрібно встановити python-пакет NumPy. NumPy – розширення мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами:

```
sudo pip3 install numpy
```

Установка *numpy* займає тривалий час без візуальних ознак роботи системи – це не зависання.

Крок 3. Завантаження та розпакування початкових файлів

Створимо в папці */home/pi* робочу папку *opencv* і перейдемо в неї за допомогою наступних команд:

```
cd /home/pi
```

```
mkdir opencv
```

```
cd opencv
```

Тепер необхідно завантажити архіви з вихідними кодами бібліотеки і додаткових модулів (рекомендується використовувати однакові версії бібліотеки і додаткових модулів) і розпакувати їх командою ***wget***:

```
wget https://github.com/opencv/opencv/archive/3.4.0.zip -O  
opencv_source.zip  
unzip opencv_source.zip
```

Оскільки є можливість використання додаткових модулів, те ж саме доведеться зробити і для них:

```
wget https://github.com/opencv/opencv_contrib/archive/3.4.0.zip -O
opencv_contrib.zip
```

```
unzip opencv_contrib.zip
```

Після того, як файли будуть розархівовані, їх можна видалити:

```
rm opencv_source.zip
```

```
rm opencv_contrib.zip
```

Створюємо папку для конфігурації:

```
cd /home/pi/opencv/opencv-3.4.0
```

```
mkdir build
```

```
cd build
```

Крок 4. Налаштування конфігурації

На даному етапі необхідно налаштувати параметри збірки, за допомогою утиліти **cmake**, яка була встановлена на початку. Для цього, всі значущі параметри передаються в якості змінних утиліти, разом з присвоєнням значень:

```
cmake -D CMAKE_BUILD_TYPE=RELEASE \
cmake -D CMAKE_INSTALL_PREFIX=/usr/local \
cmake -D INSTALL_C_EXAMPLES=OFF \
cmake -D INSTALL_PYTHON_EXAMPLES=ON \
cmake -D OPENCV_EXTRA_MODULES_PATH=/home/pi/opencv
/opencv_contrib-3.4.0/modules \
cmake -D BUILD_EXAMPLES=ON \
cmake -D BUILD_DOCS=ON \
cmake -D ENABLE_NEON=ON ..
```

Якщо створення конфігурації завершилося повідомленнями *Configuring done* і *Generating done* (рисунок 4.29), то настройка параметрів пройшла успішно.

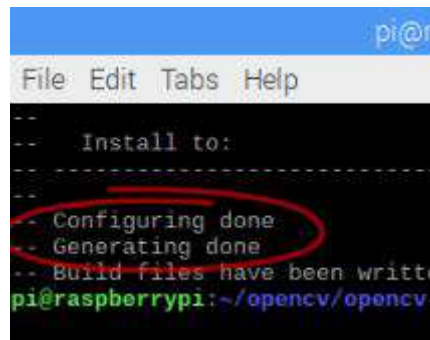


Рисунок 4.29 – Повідомлення про успішне завершення створення конфігурації параметрів

Тут же, в звіті, перед початком складання бінарників, потрібно переконаватися, що додаткові модулі (наприклад *face*) також будуть включені в збірку:

--OpenCV modules:

--To be built: aruco bgsegm bioinspired calib3d ccalib core datasets dnn dpm face features2d flann freetype fuzzy highgui img_hash imgcodecs imgproc line_descriptor ml objdetect optflow phase_unwrapping photo plot python2 python3 python_bindings_generator reg rgbd saliency shape stereo stitching structured_light superres surface_matching text tracking ts video videoio videostab xfeatures2d ximgproc xobjdetect xphoto

--Disabled: js world

--Disabled by dependency: -

--Unavailable: cnn_3dobj cudaarithm cudabgsegm cudacodec cudafeatures2d cudafilters cudaimgproc cudalegacy cudaobjdetect cudaoptflow cudastereo cudawarping cudev cvv dnn_modern hdf java matlab ovis sfm viz

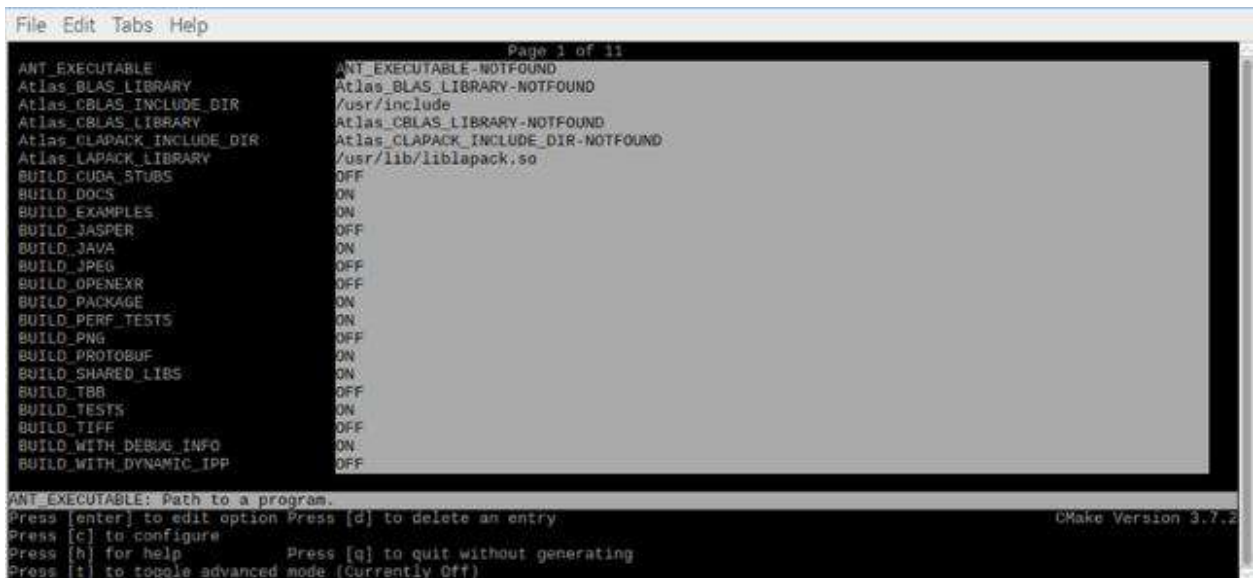
--Applications: tests perf_tests examples apps

--Documentation: NO

--Non-free algorithms: NO

Після завершення конфігурації, її настройки можна змінити за допомогою утиліти, що надає графічний інтерфейс (рисунок 4.30):

csmake ..

Рисунок 4.30 - Графічний інтерфейс *cmake*

Крок 5. Компіляція бібліотеки *OpenCV*

Після настройки конфігурації, можна приступати до складання. Параметр *-jn*, задає кількість задіяних ядер процесора (*n*) – для Raspberry Pi 3 можна задіяти 4 ядра:

```
make -j4
```

В процесі компіляції процесор дуже сильно гріється. При відсутності радіатора процесор буде знижувати тактову частоту, що дуже ймовірно призведе до зависання системи.

З метою усунення можливості подібних проблем, рекомендується використовувати меншу кількість ядер, наприклад 1 або 2 (*-j1* або *-j2*).

У разі зависання, рекомендується, після перезавантаження, відновити процедуру компіляції із залученням одного ядра:

```
cd /home/pi/opencv/opencv-3.4.0/build
```

```
make -j1
```

Крок 6. Установка бібліотеки *OpenCV*

Після успішної компіляції, бібліотеку OpenCV необхідно встановити. Робиться це за допомогою тієї ж утиліти *make*:

```
sudo make install
```

Після того, як бібліотека OpenCV інстальована, утилітою *ldconfig*, створюємо необхідні зв'язки і формуємо кеш встановлених динамічних бібліотек:

```
sudo ldconfig
```

Для перевірки, в оболонці Shell (Python 3) можна внести пару рядків:

```
>>> import cv2
>>> cv2.__version__
```

Як результат виконання команд Python видасть версію бібліотеки OpenCV.

Крок 7. Видалення початкових кодів

Після того як бібліотека встановлена, всі вихідні файли можна видалити (папка *opencv* – приблизно 6 Гб) наступними командами:

```
cd ~
rm -rf opencv
```

4.8 Установка бібліотек для роботи з GPIO

Бібліотека RPi.GPIO написана спеціально для управління портами GPIO на мові Python. Ці порти дозволяють збирати дані з датчиків, управляти двигунами, ключами, реле та іншими засобами за допомогою програми. Опис призначення всіх 40 pin порту GPIO описано в розділі 3.3.

При роботі з портом GPIO необхідно бути дуже уважним, великий відсоток «згорілих» Raspberі Pi були з вини користувачів, що неправильно зробили підключення.

На першому кроці необхідно завантажити архів з офіційного Web-сайту [sourceforge.net](https://sourceforge.net/projects/raspberry-gpio-python/files/?Source=navbar) ([https://sourceforge.net/projects/raspberry-gpio-python/files/? Source = navbar](https://sourceforge.net/projects/raspberry-gpio-python/files/?Source=navbar)) [322]. На сторінці можна побачити велику кількість версій бібліотеки, необхідно обрати саму останню версію. На рисунку 4.31 представлений скріншот сайту з версією бібліотеки 0.7.0. Архів буде завантажений в папку */home/pi/Downloads*.

Name ▾	Modified ▾	Size ▾	Downloads / Week ▾
📁 raspbian-buster	2019-07-22		18 📄
📁 raspbian-stretch	2018-11-16		1 📄
📁 old releases	2018-11-16		29 📄
📁 raspbian-jessie	2016-10-30		0 📄
📁 raspbian-wheezy	2016-02-29		0 📄
RPi.GPIO-0.7.0.tar.gz	2019-07-22	30.8 kB	64 📄 ⓘ
Totals: 6 Items		30.8 kB	112 📄

Рисунок 4.31 – Скріншот Web-сайту sourceforge.net з архівом бібліотекою RPi.GPIO-0.7.0.tar.gz

На другому кроці необхідно перейти в папку Downloads, де знаходиться завантажений архів, натиснути праву кнопку миші і обрати у спливаючому меню Extract Here, як показано на рисунку 4.32 і розархівувати його в дану папку.

Ця команда розпаковує бібліотеку в папку, в яку поміщений архів. Поруч з ним буде додана нова папка з назвою RPi.GPIO-0.7.0.

Третім кроком потрібно встановити *python-dev* за допомогою терміналу і команди:

sudo apt-get install python-dev

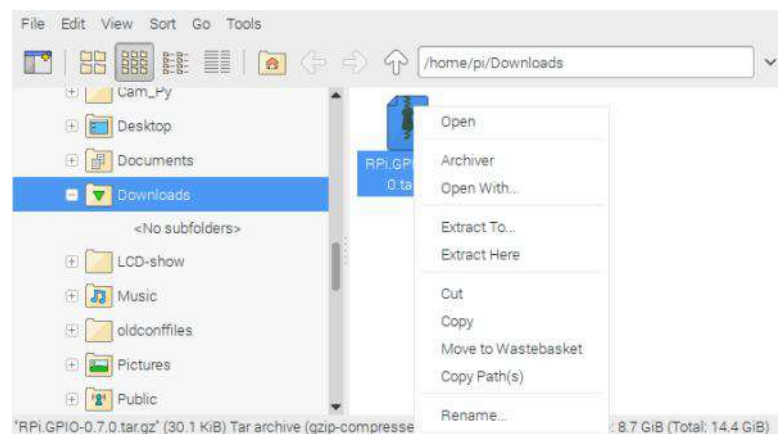


Рисунок 4.32 – Розпаковка архіву бібліотеки RPi.GPIO-0.7.0.tar.gz

В ході установки буде поставлено питання «Do you want to continue? [Y/n]» (рисунок 4.33), потрібно натиснути « Yes » і Enter.

```
File Edit Tabs Help
pi@raspberrypi:~ $ sudo apt-get install python-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  libpython-dev libpython2.7-dev python2.7-dev
The following NEW packages will be installed:
  libpython-dev libpython2.7-dev python-dev python2.7-dev
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 18.2 MB of archives.
After this operation, 25.7 MB of additional disk space will be used.
Do you want to continue? [Y/n] 
```

Рисунок 4.33 – Установка *python-dev*

Підтвердженням коректної установки буде повідомлення про успішну установки у вікні терміналу, як показано на рисунку 4.34.

```
File Edit Tabs Help
Get:3 http://mirrordirector.raspbian.org/raspbian/ jessie/main libpython-dev arm
hf 2.7.9-1 [19.6 kB]
Get:4 http://mirrordirector.raspbian.org/raspbian/ jessie/main python2.7-dev arm
hf 2.7.9-2+deb8u1 [287 kB]
Fetched 18.2 MB in 49s (367 kB/s)
Selecting previously unselected package libpython2.7-dev:armhf.
(Reading database ... 121836 files and directories currently installed.)
Preparing to unpack .../libpython2.7-dev_2.7.9-2+deb8u1_armhf.deb ...
Unpacking libpython2.7-dev:armhf (2.7.9-2+deb8u1) ...
Selecting previously unselected package libpython-dev:armhf.
Preparing to unpack .../libpython-dev_2.7.9-1_armhf.deb ...
Unpacking libpython-dev:armhf (2.7.9-1) ...
Selecting previously unselected package python2.7-dev.
Preparing to unpack .../python2.7-dev_2.7.9-2+deb8u1_armhf.deb ...
Unpacking python2.7-dev (2.7.9-2+deb8u1) ...
Selecting previously unselected package python-dev.
Preparing to unpack .../python-dev_2.7.9-1_armhf.deb ...
Unpacking python-dev (2.7.9-1) ...
Processing triggers for man-db (2.7.0.2-5) ...
Setting up libpython2.7-dev:armhf (2.7.9-2+deb8u1) ...
Setting up libpython-dev:armhf (2.7.9-1) ...
Setting up python2.7-dev (2.7.9-2+deb8u1) ...
Setting up python-dev (2.7.9-1) ...
pi@raspberrypi:~ $ 
```

Рисунок 4.34 – Підтвердження коректної установки *python-dev*

Для установки бібліотеки RPi.GPIO, необхідно в даному терміналі перейти в папку (за замовчуванням /home/pi/Downloads/RPi.GPIO-0.7.0), в якій був розпакований файл бібліотеки RPi.GPIO-0.7.0.tar.gz ,за допомогою простої консольної команди:

```
cd /home/pi/Downloads/RPi.GPIO-0.7.0
```

і запустити установку бібліотеки за допомогою команди

```
sudo python setup.py install
```

Після успішної установки на екрані з'явиться новий звіт, як представлено на рисунку 4.35.

```

relro -Wdate-time -D_FORTIFY_SOURCE=2 -g -fdebug-prefix-map=/build/python2.7-Ub3
vap/python2.7-2.7.13=. -fstack-protector-strong -Wformat -Werror=format-security
build/temp.linux-armv7l-2.7/source/py_gpio.o build/temp.linux-armv7l-2.7/source
/c_gpio.o build/temp.linux-armv7l-2.7/source/cpuinfo.o build/temp.linux-armv7l-2
.7/source/event_gpio.o build/temp.linux-armv7l-2.7/source/soft_pwm.o build/temp.
linux-armv7l-2.7/source/py_pwm.o build/temp.linux-armv7l-2.7/source/common.o bui
ld/temp.linux-armv7l-2.7/source/constants.o -o build/lib.linux-armv7l-2.7/RPi/_G
PIO.so
running install_lib
creating /usr/local/lib/python2.7/dist-packages/RPi
copying build/lib.linux-armv7l-2.7/RPi/_GPIO.so -> /usr/local/lib/python2.7/dist
-packages/RPi
copying build/lib.linux-armv7l-2.7/RPi/__init__.py -> /usr/local/lib/python2.7/d
ist-packages/RPi
creating /usr/local/lib/python2.7/dist-packages/RPi/GPIO
copying build/lib.linux-armv7l-2.7/RPi/GPIO/__init__.py -> /usr/local/lib/python
2.7/dist-packages/RPi/GPIO
byte-compiling /usr/local/lib/python2.7/dist-packages/RPi/__init__.py to __init
_.pyc
byte-compiling /usr/local/lib/python2.7/dist-packages/RPi/GPIO/__init__.py to __
init__.pyc
running install_egg_info
Writing /usr/local/lib/python2.7/dist-packages/RPi.GPIO-0.7.0.egg-info
pi@raspberrypi:~/Downloads/RPi.GPIO-0.7.0 $

```

Рисунок 4.35 – Приклад звіту про успішну установки бібліотеки RPi.GPIO-0.7.0

Крім вищеописаної бібліотеки RPi.GPIO існують і інші. Для доступу до GPIO контактів використовується також бібліотека WiringPi [323-325]. Ця бібліотека написана на мові C, використовується в середовищах C, C ++ і

RTB (BASIC) та інших. Вона створювалася ідентичною до бібліотеки Wiring на Arduino.

5 ПІДКЛЮЧЕННЯ ТА ПРОГРАМУВАННЯ ДАТЧИКІВ МОВОЮ PYTHON

5.1 Підключення ультразвукового датчика HC-SR04

Особливістю роботи даного датчика є подвоєння відстані до об'єкта. Тобто, після запуску вимірювання та відправки на пін Trig імпульсу в 10 мкс, сенсор генерує таку ж саму серію імпульсів, чекає відповіді і потім відправляє на пін Echo сигнал, тривалість якого дорівнює часу, потрібному на отримання відправленого сигналу.

HC-SR04 датчик має 4 піна:

- VCC – живлення сенсора 5 В (напрямку з піна номер 2 Raspberry);
- Grd – пін 6 на Raspberry Pi;
- Trig. – на цей пін необхідно надіслати сигнал тривалістю 10 мкс для того щоб сенсор спрацював. Сенсор посилає звуковий ряд з 8 імпульсів в 40кГц і приймає їх;
- Echo з цього піна зчитується інформація з датчика [326].

З цих даних можна порахувати на скільки часто можна отримувати дані про найближчі об'єкти для цього необхідно врахувати:

- мінімальна затримка між вимірами повинна бути 50 мс;
- 10 мкс на генерацію збудливого сигналу (можна знехтувати);
- $1/340$ секунди = 3мс на вимір максимально доступної відстані.

Припустимо, максимально доступна відстань становить 500 см. Звук буде проходити в обох напрямках, тобто йому потрібно пройти 1 м. Швидкість звуку приблизно 340 мс. Звуковий сигнал пройде цей метр за $1/340$ с, або приблизно 3мс.

- 3 мс – час потрібний датчику щоб передати сигнал на пін Echo.

В підсумку. $50 + 3 + 3 = 56$ мс. Це мінімально дозволені межі одержуваної інформації про відстань до об'єкту. Наприклад, якщо MP

рухається зі швидкістю 20 км/год це дорівнює 5,5 мс, тобто за зазначений час 56 мс він пройде 30 см. Якщо швидкості не високі, точності вимірювань буде достатньою. Схема підключення датчика HC-SR04 представлений на рис. 5.1.

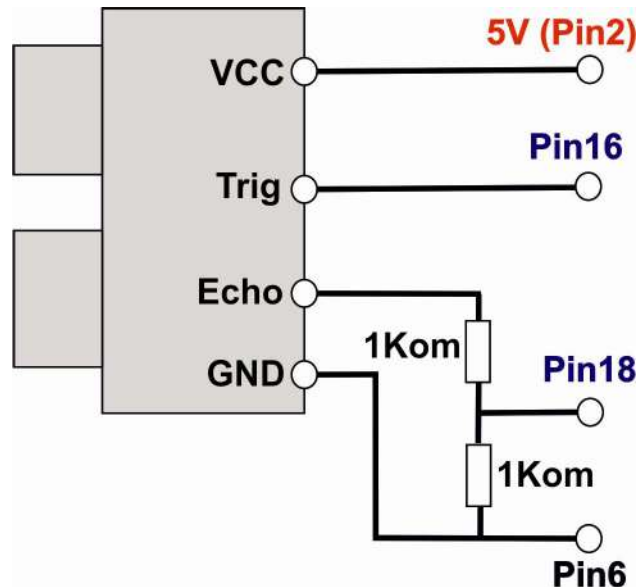


Рисунок 5.1 – Схема підключення датчика HC-SR04

Додаткова особливість. Живлення схеми йде від 5 В і вихідна напруга з Echo Піна – 5В, тому потрібно її зменшити перед подачею на вхідний пін Raspberry Pi. Це можна зробити найпростішим дільником напруги, який складається з двох резисторів. У схемі представлений на рисунку 5.1 є два резистора в 1 кОм. Знімаючи сигнал з одного з них отримуємо 2,5 вольтовий сигнал, що є достатнім для розпізнавання сигналу. Мінімальний рівень сигналу для визначення наявності сигналу і визначення його як High – 1,3 В, максимальний рівень дорівнює 3,3 В. Тому 2,5 В – більш ніж достатньо.

Для написання програми з обробки інформації з ультразвукового датчика HC-SR04 необхідно підключити дві бібліотеки:

```
import Rpi.GPIO as GPIO
```

```
import time
```

У коді визначаємо два піна. Один як вхідний, інший як вихідний. З останнього будемо відправляти імпульс для збудження датчика, а з першого буде йти зчитування інформації з сенсора.

```

GPIO.setmode(GPIO.BOARD)
TRIG = 16
ECHO = 18
GPIO.setup(TRIG, GPIO.OUT, initial=0)
GPIO.setup(ECHO, GPIO.IN)

```

Далі необхідно почекати мінімум 50 мс – мінімальна затримка між вимірами. В цьому скрипті вона не особливо потрібна. Але якщо використовувати цей скрипт як модуль у великій програмі, яка буде його викликати, то краще використовувати цю затримку. Також у даному випадку при цьому краще використовувати зацикленість вимірювань відстані.

Після цього необхідно надіслати сигнал на пін Trig сенсора. Тривалість сигналу 10 мкс:

```

GPIO.output(TRIG,1)
time.sleep(0.00001)
GPIO.output(TRIG,0)

```

Далі необхідно чекати поки не почне надходити сигнал на порт Echo. Потрібно відзначити потрібний проміжок часу сигналу. Як тільки сигнал почне надходити, необхідно зрозуміти коли він закінчиться. Як тільки він перерветься необхідно визначити коли він закінчився. Робиться це за допомогою змінних start та stop та модуля часу time.

```

while GPIO.input(ECHO) == 0:
    pass
start = time.time()
while GPIO.input(ECHO) == 1:
    pass
stop = time.time()

```

Знаючи початковий і кінцевий час сигналу можна знайти його тривалість і отже відстань пройдену сигналом. Не забуваємо, що сигнал проходить подвійну відстань і результат (Відстань = Швидкість / Час) потрібно ділити на 2:

```
print "Distance = ",(stop - start) * 17000,"sm"
```

Приклад програмного коду на наведений в додатку И.

5.2 Підключення та програмування датчика температури і вологості DHT22

DHT22 – це універсальний і недорогий датчик вологості, який також може розраховувати температуру робочого середовища МР. Цей датчик має відносно велику відстань передачі, що дозволяє датчику передавати дані по дротах на відстані до 20 м від Raspberry Pi [327].

DHT22 – це цифровий датчик з вбудованим аналого-цифровим перетворювачем. Конвертор значно полегшує підключення датчика до Raspberry Pi, оскільки не потрібно мати будь-яких додаткових чипів. Найбільшим недоліком датчиків DHT11 та DHT22 є те, що вони є досить повільними датчиками. Вони мають частоту дискретизації один раз на секунду для DHT11 та один раз на кожні 2 секунди для DHT22.

DHT22 – це універсальний і недорогий датчик вологості. Варто зауважити, що підключення DHT22 до Raspberry Pi є аналогічним з датчиками вологості DHT11 і AM2302, оскільки всі вони використовують однакові терморегулятори.

Щоб спростити складання ланцюга датчика вологості, можна включили терморегулятори датчика DHT22. Терморегулятори датчика DHT22 наведено на рисунку 5.2.

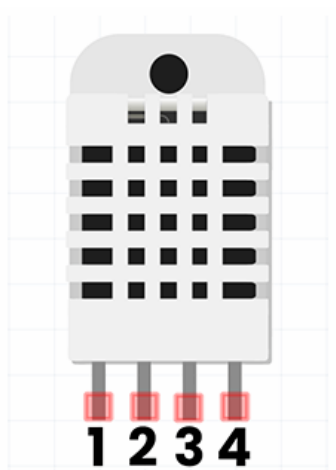


Рисунок 5.2 – Розпіновка датчика DHT22

Опишемо кожен пін і його призначення:

- пін 1 – це VCC (джерело живлення);
- пін 2 – це дані (сигнал даних);
- пін 3 – недійсний (не з'єднувати);
- пін 4 – Gnd;

Для підключення датчика DHT22 потрібно виконати наступний алгоритм підключення:

- помістіть резистор 10 кОм між контактом 1 і контактом 2 DHT22;
- підключіть контакт 1 DHT22 до фізичного контакту 1 (3v3) на Rpi;
- підключіть контакт 2 DHT22 до фізичного контакту 7 (GPIO4) на Rpi;
- підключіть контакт 4 DHT22 до фізичного контакту 6 (GND) на Rpi.

Перш ніж почати програмувати скрипт для датчика вологості Raspberry Pi, необхідно оновити Raspberry Pi. Для цього потрібно виконати наступні дві команди:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

Встановіть обидва python3-dev і python3-pip, виконавши команду нижче:

```
sudo apt-get install python3-dev python3-pip
```

Перш ніж продовжити установку бібліотеки DHT, необхідно виконати наступну команду, щоб переконатися, що є останні версії пакетів `setuptools`, `wheel` і `pip python`:

```
sudo python3 -m pip install --upgrade pip setuptools wheel
```

Тепер, використовуючи `pip`, проведемо установку DHT-бібліотеку Adafruit на Raspberry Pi. Дана бібліотека Python дозволить реалізувати взаємодію з датчиком DHT22.

Бібліотека також підтримує датчики вологості / температури DHT11 і AM2302, що робить її незамінною для вивчення того, як її використовувати.

Виконайте наступну команду, щоб встановити бібліотеку DHT на Raspberry Pi:

```
sudo pip3 install Adafruit_DHT
```

У цьому файлі потрібно ввести такі рядки коду. Нижче кожен розділ коду буде прокоментований і пояснений у міру написання скрипта:

```
import Adafruit_DHT
```

Цей рядок імпортує бібліотеку «Adafruit_DHT», яка була інстальована за допомогою `pip`.

Дана бібліотека буде використана для взаємодії з датчиком DHT22. Бібліотека дозволяє дуже легко отримати інформацію про температуру і вологість від датчика за допомогою кількох рядків коду Python:

```
DHT_SENSOR = Adafruit_DHT.DHT22
```

У цьому рядку визначено константу для датчика, яку потрібно використовувати для бібліотеки DHT. У бібліотеці Adafruit DHT є три варіанти вибору, в залежності від типу використовуваного датчика.

В даному навчальному посібнику використано DHT22, тому визначим нашу константу «DHT_SENSOR» як «Adafruit_DHT.DHT22».

Потім визначаємо константу, яка називається «DHT_PIN», що призначена для зберігання номера контакту GPIO, до якого підключений DHT22 на Raspberry Pi:

```
DHT_PIN = 4
```

Використане число відноситься до нумерації BCM виводів GPIO. В даному випадку це означає, що потрібно використовувати число «4», так як буде використовуватися вивод GPIO4 для з'єднання та передачі даних. Якщо буде використовуватися інший вивод GPIO, переконайтеся, що його номер буде вказано відповідним чином:

```
while True:
```

```
    humidity, temperature = Adafruit_DHT.read_retry(DHT_SENSOR,  
DHT_PIN)
```

На початку кожного циклу бібліотека Adafruit DHT видаляє отриману інформацію про вологість та температуру з датчика DHT22, через константу DHT_SENSOR і DHT_PIN.

Функція «read_retry» буде постійно намагатися отримати дані від датчика вологості:

```
if humidity is not None and temperature is not None:
```

```
    print("Temp={0:0.1f}*C    Humidity={1:0.1f}%".format(temperature,  
humidity))
```

```
else:
```

```
    print("Failed to retrieve data from humidity sensor")
```

Необхідно перевірити, щоб бібліотека DHT повертала необхідні дані.

Якщо змінні «humidity» або «temperature» рівні «None», то буде повертатися повідомлення, яке повідомить про неможливість отримати дані з датчика вологості.

Якщо є значення обох цих змінних, то бачимо повідомлення, що показує температуру і вологість отримані з датчика. Повний текст скрипта наведено нижче:

```
import Adafruit_DHT
```

```
DHT_SENSOR = Adafruit_DHT.DHT22
```

```
DHT_PIN = 4
```

```
while True:
```

```
humidity, temperature = Adafruit_DHT.read_retry(DHT_SENSOR,
DHT_PIN)
```

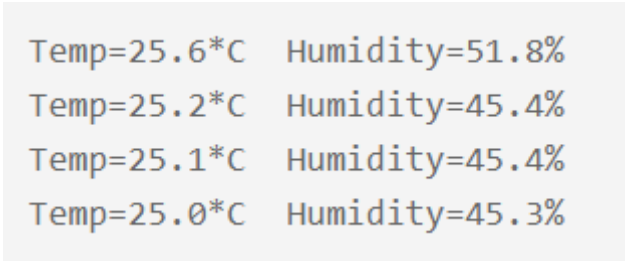
if humidity is not None and temperature is not None:

```
print("Temp={0:0.1f}*C    Humidity={1:0.1f}%"format(temperature,
humidity))
```

else:

```
print("Failed to retrieve data from humidity sensor")
```

При виконанні наведеного вище скрипта відобразиться температура і вологість в командному рядку. Приклад результату роботи скрипта наведені на рисунку 5.3.



```
Temp=25.6*C    Humidity=51.8%
Temp=25.2*C    Humidity=45.4%
Temp=25.1*C    Humidity=45.4%
Temp=25.0*C    Humidity=45.3%
```

Рисунок 5.3 – Відображення значень температура і вологість в командному рядку

Нижче наведемо приклад скрипта для збору інформації з датчика DHT22 та запису аналізу в файл. Файл зможе бути прочитаний як Microsoft Excel або Google Spreadsheets, що дозволить реалізувати контроль за робочим оточенням МР.

Вдосконалюємо вище наведений скрипт за допомогою додаванням нових рядків програмного коду. Нижче наводяться тільки нові доповнення до існуючого скрипту:

```
import os
```

```
import time
```

```
import Adafruit_DHT
```

Додамо бібліотеки «os» і «time», які будуть перевіряти існування CSV-файл, перш ніж проводити записати в нього даних з датчика. У

бібліотеці «time» необхідно встановлювати мітки часу в кожному новому рядку з поточною датою і часом коли були отримані дані з датчика:

```
DHT_SENSOR = Adafruit_DHT.DHT22
```

```
DHT_PIN = 4
```

Використовуємо фрагмент скрипта з попереднього прикладу і додамо новий фрагмент:

```
try:
```

```
    f = open('/home/pi/humidity.csv', 'a+')
```

```
    if os.stat('/home/pi/humidity.csv').st_size == 0:
```

```
        f.write('Date,Time,Temperature,Humidity\r\n')
```

```
except:
```

```
    pass
```

Реалізуємо все в операторі *try* для обробки будь-яких можливих помилок. Реалізуємо відкриття дескриптора файлу «/home/pi/humidity.csv» з застосуванням тегом «+». Тег + означає, що будь-які дані, записані в файл, будуть додані в його кінець.

Далі використовуємо бібліотеку «os», щоб дізнатися, чи вносились записи будь-коли в цей файл раніше. Якщо розмір повертається як 0, записуємо початкову рядок в файл. Цей рядок буде містити заголовки наших стовпців, щоб користувачеві було простіше аналізувати дані.

```
while True:
```

```
    humidity, temperature = Adafruit_DHT.read_retry(DHT_SENSOR,
DHT_PIN)
```

Цикл *while* дозволяє реалізувати використання бібліотеки DHT для повторного зчитування даних з датчика вологості DHT22.

Нижче наведений фрагмент скрипта у вигляді функції, що дозволяє зберігати зчитані з датчика дані, в змінні «*humidity*» і «*temperature*»:

```
if humidity is not None and temperature is not None:
```

```
    f.write('{0},{1},{2:0.1f}*C,{3:0.1f}%\r\n'.format(time.strftime('%m/%d/%y'),
time.strftime('%H:%M'), temperature, humidity))
```


else:

```
print("Failed to retrieve data from humidity sensor")
```

```
time.sleep(30)
```

Дана зміна в цьому фрагменті коду означає, що замість виведення температури і вологості на консоль робляться записи в файл «*humidity.csv*» за допомогою дескриптора файлу, який був відкритий раніше. Далі проводимо форматування тексту, записаного в файл, щоб включити поточну дату, поточний час і відформатовану температуру і вологість з датчика. В кінці циклу встановлюємо час виконання сценарію на 30 секунд. Якщо необхідна зміна часу спрацювання датчика, потрібно змінити це число залежно від вимог.

Доповнений скрипт має наступний загальний вид:

```
import os
```

```
import time
```

```
import Adafruit_DHT
```

```
DHT_SENSOR = Adafruit_DHT.DHT22
```

```
DHT_PIN = 4
```

```
try:
```

```
    f = open('/home/pi/humidity.csv', 'a+')
```

```
    if os.stat('/home/pi/humidity.csv').st_size == 0:
```

```
        f.write('Date,Time,Temperature,Humidity\r\n')
```

```
except:
```

```
    pass
```

```
while True:
```

```
    humidity, temperature = Adafruit_DHT.read_retry(DHT_SENSOR,  
DHT_PIN)
```

```
    if humidity is not None and temperature is not None:
```

```
        f.write('{0},{1},{2:0.1f}*C,{3:0.1f}%\r\n'.format(time.strftime('%m/%d/%y'),  
time.strftime('%H:%M'), temperature, humidity))
```

else:

```
print("Failed to retrieve data from humidity sensor")
```

```
time.sleep(30)
```

На рисунку 5.4 наведено приклад вмісту файлу «*humidity.csv*», який має наступну архітектуру.

```
Date,Time,Temperature,Humidity
05/04/19,05:47,22.3*С,50.6%
05/04/19,05:48,22.2*С,50.4%
05/04/19,05:48,22.2*С,50.6%
05/04/19,05:49,22.2*С,50.4%
05/04/19,05:49,22.2*С,50.4%
05/04/19,05:50,22.2*С,50.3%
05/04/19,05:50,22.2*С,50.3%
05/04/19,05:51,22.2*С,50.2%
```

Рисунок 5.4 – Вміст файлу «*humidity.csv*»

5.3 Підключення та програмування управління сервоприводами для системи комп'ютерного зору

Для управління поворотом камери на МР потрібно використовувати сервопривід моделі TowerPro SG90 [328, 330]. Загальний вигляд сервоприводу представлена на рисунку 5.5.



Рисунок 5.5 – Загальний вигляд сервоприводу моделі TowerPro SG90

Параметри сервоприводу моделі TowerPro SG90:

- робоча напруга: 4.8 DC;
- швидкість: 60 градусів за 0.1с;
- крутний момент: 1.6 кг см;
- діапазон ширини імпульсу: 0.5 мс - 2.5 мс.

Підключення проводиться наступним способом: від сервоприводу йде шлейф з трьох проводів: червоний – живлення підключається до контакту 5V (пін 2) або безпосередньо до джерела живлення; коричневий або чорний – земля GND (пін 6 Raspberry Pi).

Дане серво не може обертатися на 360 градусів. Воно має нейтральну позицію (Neutral) і межі від 0 да 180 градусів. Для встановлення серво в нейтральну позицію, необхідно подати високий сигнал тривалістю 1.5 мс, в 0 градусів – 0.5 мс, в 180 градусів – 2.5 мс (рис. 5.6).

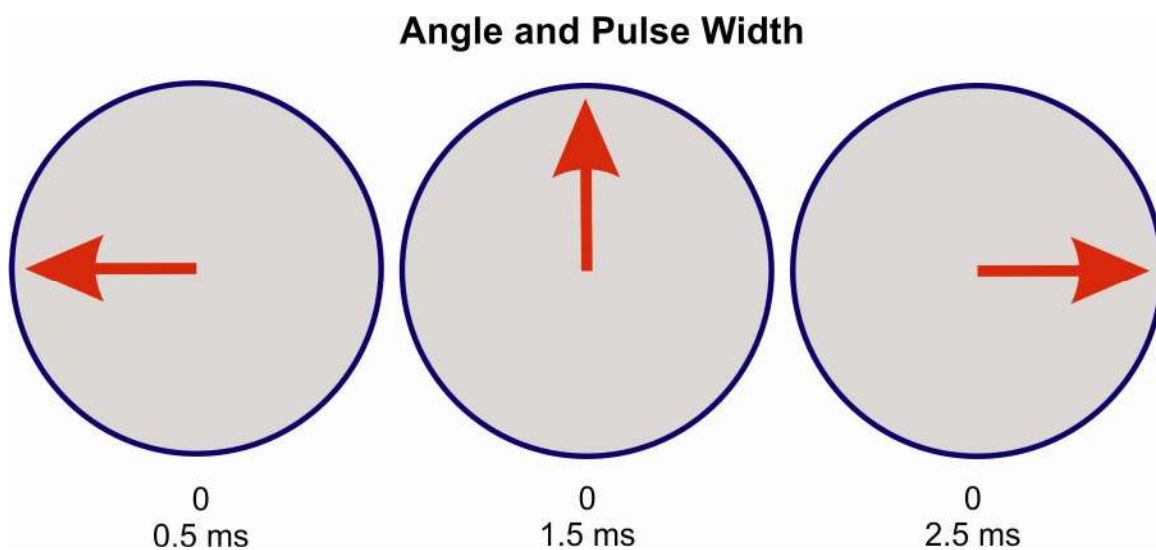


Рисунок 5.6 – Тривалість високого сигналу на кут повороту сервоприводу

Raspberry Pi не має аналогового виходу, але можема імітувати його, використовуючи підхід ШІМ. Можливо робити цифровий сигнал з фіксованою частотою, де змінємо ширину послідовності імпульсів, як наведено на рисунку 5.7.

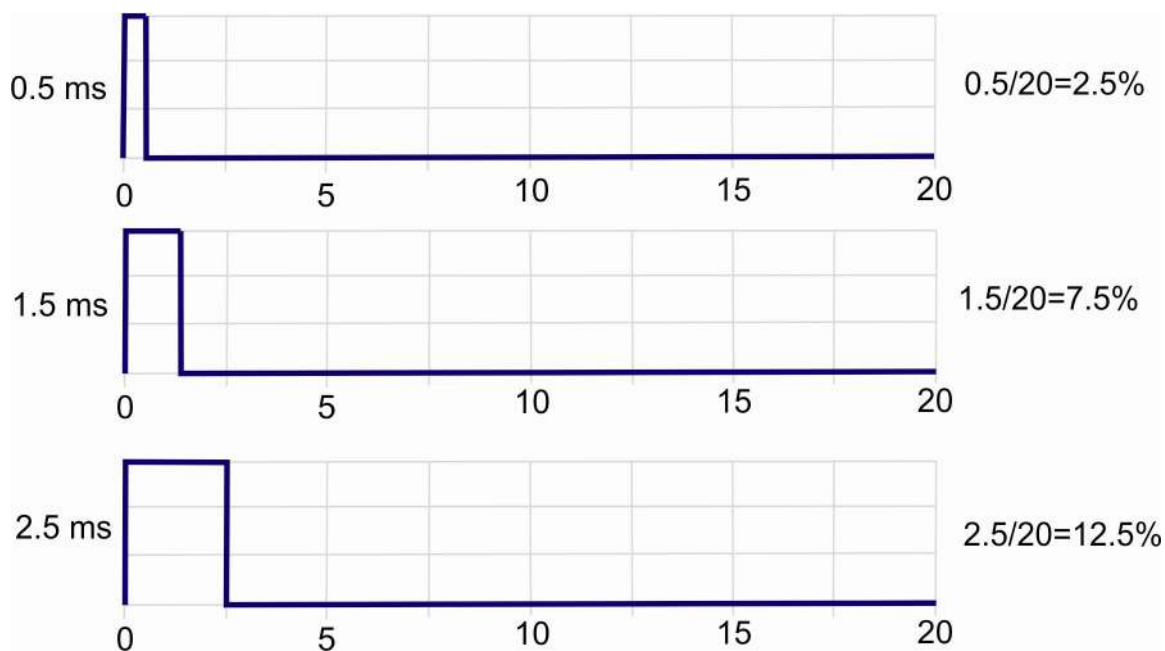


Рисунок 5.7 – Графік залежності довжини імпульсів від положення сервоприводу

Напишем наступний скрипт мовою Python:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
GPIO.setup(7, GPIO.OUT)
p = GPIO.PWM(7, 50)
p.start(7.5)

try:
    while True:
        p.ChangeDutyCycle(7.5)
        time.sleep(1)
        p.ChangeDutyCycle(12.5)
        time.sleep(1)
        p.ChangeDutyCycle(2.5)
        time.sleep(1)
```

except KeyboardInterrupt:

p.stop()

GPIO.cleanup()

Наведений скрипт дозволяє управляти сервоприводом через сьомий пін, за допомогою ШІМ, з частотою 50 Герц: GPIO.PWM (7,50). У додатку наведено скрипт управління двома сервоприводами, для забезпечення горизонтального і вертикального кута повороту камери.

6. РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ МОБІЛЬНОЮ ПЛАТФОРМОЮ НА БАЗІ МОВИ PYTHON

6.1 Скрипт управління DC двигунами мобільної платформи

На першому етапі тестування системи управління DC двигунами мобільної платформи 4WD напишемо програму мовою Python, без використання PWM для управління швидкістю руху [331–333].

```
import time

import RPi.GPIO as GPIO

# Підготавлюємо пини GPIO.
GPIO.cleanup()

GPIO.setmode(GPIO.BCM)

#Hidden errors
GPIO.setwarnings(False)

#колеса(all) install
GPIO.setup(4,GPIO.OUT)
GPIO.output(4,GPIO.LOW)
GPIO.setup(17,GPIO.OUT)
GPIO.output(17,GPIO.LOW)
GPIO.setup(27,GPIO.OUT)
GPIO.output(27,GPIO.LOW)
GPIO.setup(22,GPIO.OUT)
GPIO.output(22,GPIO.LOW)

#-----Forfard -----
GPIO.output(17,GPIO.HIGH)
GPIO.output(27,GPIO.HIGH)

# чекаємо 5 секунд.
time.sleep(5)
```

```

GPIO.output(17,GPIO.LOW)
GPIO.output(27,GPIO.LOW)
#----Down-----
GPIO.output(4,GPIO.HIGH)
GPIO.output(22,GPIO.HIGH)
# чекаємо 5 секунд.
time.sleep(5)
GPIO.output(4,GPIO.LOW)
GPIO.output(22,GPIO.LOW)
# ---left -----
GPIO.output(27,GPIO.HIGH)
GPIO.output(4,GPIO.HIGH)
# чекаємо 5 секунд.
time.sleep(5)
GPIO.output(27,GPIO.LOW)
GPIO.output(4,GPIO.LOW)
#----rigeht-----
GPIO.output(17,GPIO.HIGH)
GPIO.output(22,GPIO.HIGH)
# чекаємо 5 секунд.
time.sleep(5)
GPIO.output(17,GPIO.LOW)
GPIO.output(22,GPIO.LOW)
#----swich of all gpio -----
GPIO.output(17,GPIO.LOW)
GPIO.output(27,GPIO.LOW)
GPIO.output(4,GPIO.LOW)
GPIO.output(22,GPIO.LOW)

```

Широтно-імпульсна модуляція (PWM (pulse-width modulation) – процес управління потужності методом пульсуючого включення і вимикання

приладу, іншими словами це спосіб контролю живлення [334]. У даних дослідженнях ми будемо застосовувати PWM, щоб контролювати кількість струму, який проходить через мотор і, як наслідок, швидкість обертання мотора. Кожну 1/500 секунди роз'єм PWM подає імпульс. Довжина цього імпульсу контролює кількість енергії, яке отримує мотор. Якщо імпульс не надходить, мотор не буде обертатися. Якщо імпульс короткий, мотор буде обертатися повільно. Якщо імпульс довжиною в половину проміжку між імпульсами, мотор отримає половину від енергії, яку він міг би отримати, якби імпульс тривав весь проміжок між імпульсами. На рисунку 6.1 приведена діаграма сигналу PWM на Raspberry Pi.

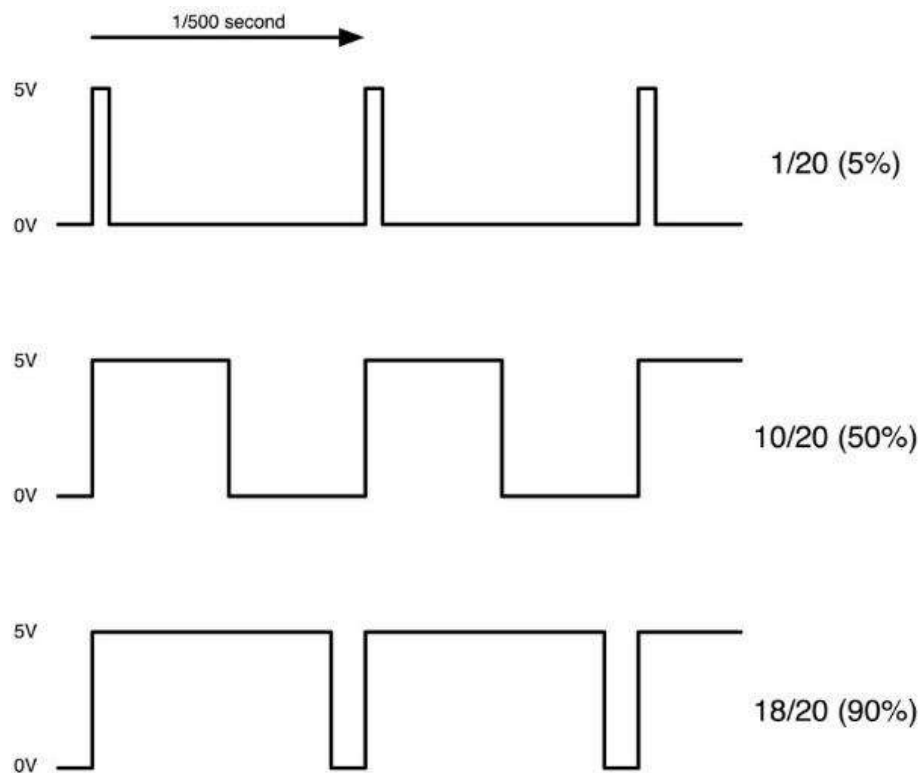


Рисунок 6.1 – Діаграма сигналу ШІМ на Raspberry Pi

6.2 Підключення і настройка відео-стріму з камери Raspbian Pi

Цей підрозділ присвячений вирішенню завдання забезпечення мобільного робота можливістю отримувати інформацію про навколишнє оточення. Для цього на початковому етапі будемо використовувати потокову

передачу відео за допомогою Raspberry Pi та камери, яка буде транслювати відео в системі реального часу на віддалену Web-сторінку. Таке рішення дозволить забезпечити доступ з будь-якого пристрою, який має браузер і підключеної до нього мережі Internet. Отже дасть можливість розробляти системи ідентифікації і розпізнання об'єктів з використанням різних математичних методів (нейронні мережі, машинне навчання і т.д.) [335-340].

Якщо використовується модуль камери Raspberry Pi, необхідно включити програмне забезпечення камери в Raspberry Pi, щоб використовувати його. У середовищі робочого столу перейдіть у вікно «Конфігурація Raspberry Pi» в меню «Налаштування», відкрийте вкладку «Інтерфейси» і включіть камеру, як показано на рисунку 6.2.

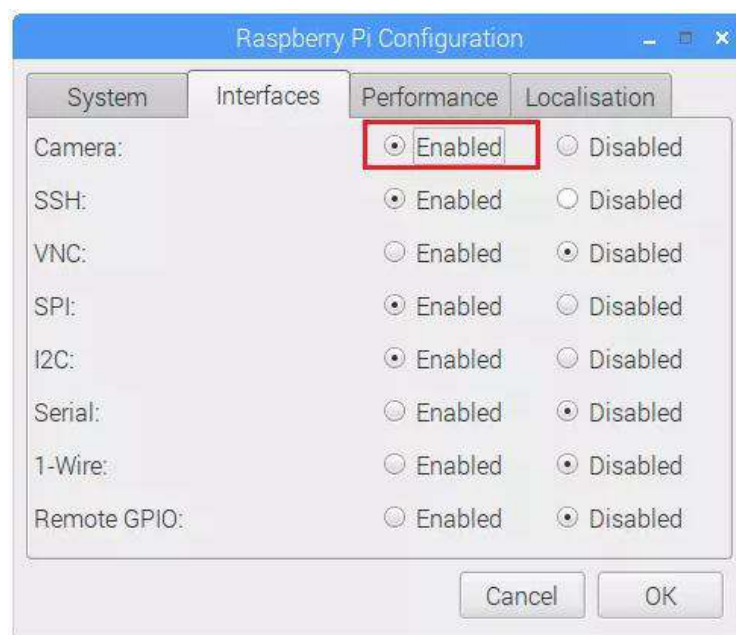


Рисунок 6.2 – Вікно Raspberry Pi Configuration

Або скористатися *Terminal*, де необхідно ввести наступну команду:

sudo raspi-config

Після роботи команди з'явиться інструмент налаштування програмного забезпечення Raspberry Pi. Необхідно обрати Interfacing Options і активізувати (Enable) P1 Camera за допомогою <Select>, як показано на рисунку 6.3.

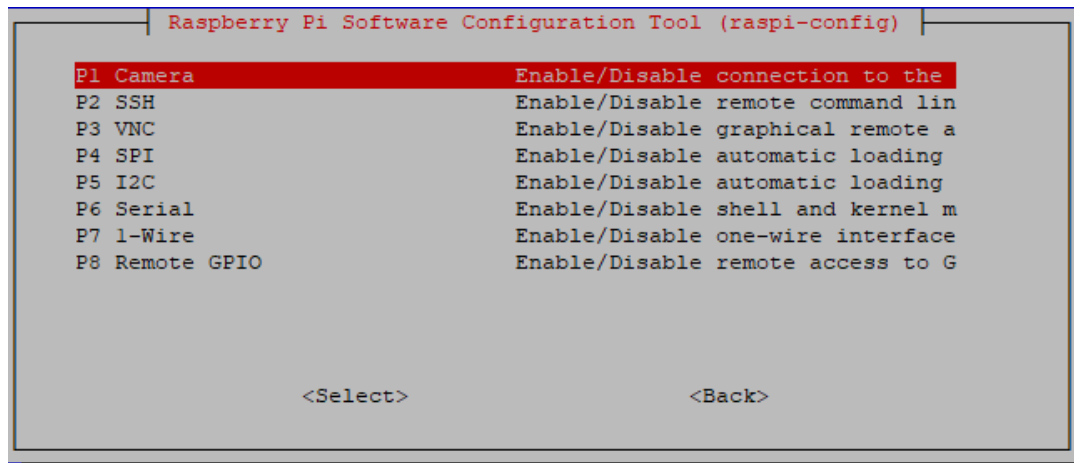


Рисунок 6.3 – Вікно настройки raspi-config

Щоб отримати доступ до веб-сервера потокового відео, необхідно знати свою IP-адресу Raspberry Pi. Для цього використовуйте наступну команду:

ifconfig

З'явиться багато інформації, включаючи поточну IP-адресу Raspberry Pi. У нашому випадку IP-адреса RPi – 192.168.1.5 (рис. 6.4).

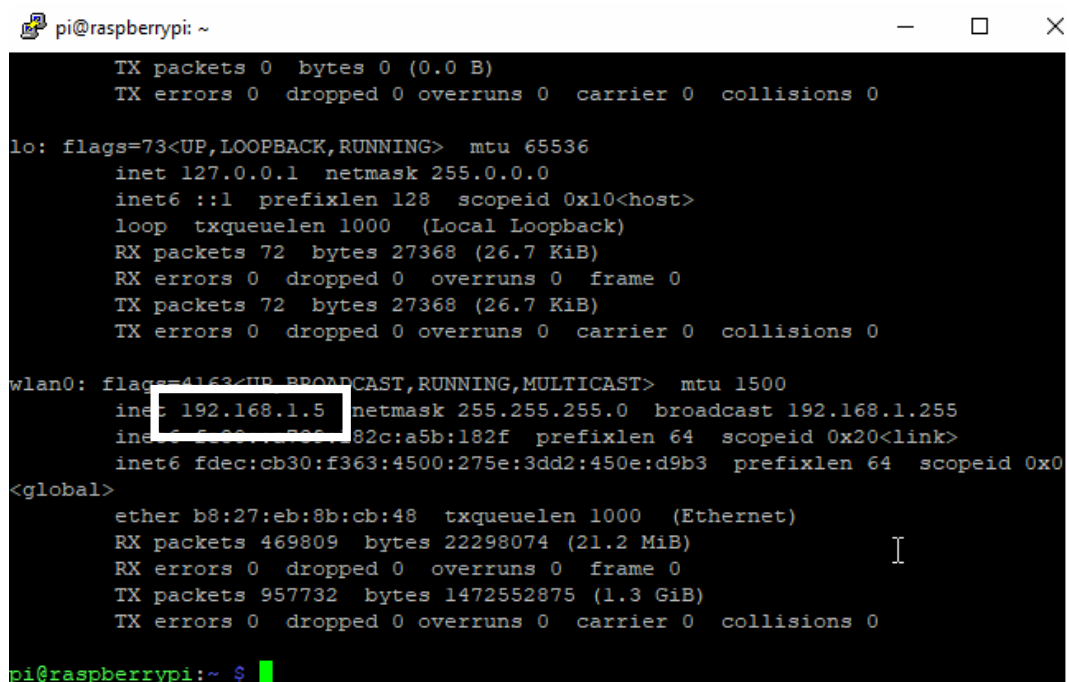


Рисунок 6.4 – Визначення IP-адреси Raspberry Pi

Підключити модуль камери Raspberry Pi дуже просто. Після відключення Pi підключіть камеру до порту Pi CSI, як показано на рис. 6.5 [341]. Переконайтесь, що камера підключена в правильній орієнтації так, щоб сині літери на стрічці були спрямовані вгору.

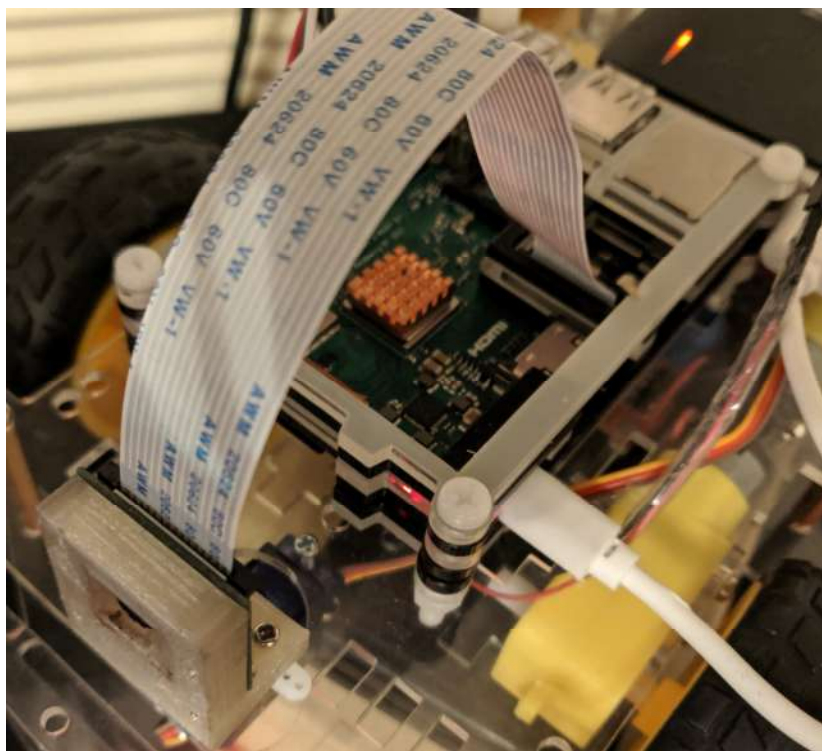


Рисунок 6.5 – Підключення модуля камери до порту CSI на Raspberry Pi

Скрипт для потокового відео буде поєднувати в собі HTML і Payten. На першому етапі створимо новий файл з ім'ям *rpi_camera_surveillance_system.py* з використанням редактора *nano* за допомогою наступної команди:

```
nano rpi_camera_surveillance_system.py
```

і ставимо в нього наступний код, що складається з 3 основних блоків.

Блок 1: Підключення бібліотек:

```
import io  
import picamera  
import logging  
import socketserver
```

```
from threading import Condition
from http import server
```

Блок 2: Блок HTML коду:

```
PAGE="""\
<html>
<head>
<title>Raspberry Pi - Surveillance Camera</title>
</head>
<body>
<center><h1>Raspberry Pi - Surveillance Camera</h1></center>
<center></center>
</body>
</html>
```

Блок 3: Блок Python коду і робота з бібліотекою *picamera*:

```
class StreamingOutput(object):
    def __init__(self):
        self.frame = None
        self.buffer = io.BytesIO()
        self.condition = Condition()
    def write(self, buf):
        if buf.startswith(b'\xff\xd8'):
            # New frame, copy the existing buffer's content and notify all
            # clients it's available
            self.buffer.truncate()
            with self.condition:
                self.frame = self.buffer.getvalue()
                self.condition.notify_all()
            self.buffer.seek(0)
        return self.buffer.write(buf)
```

```

class StreamingHandler(server.BaseHTTPRequestHandler):
    def do_GET(self):
        if self.path == '/':
            self.send_response(301)
            self.send_header('Location', '/index.html')
            self.end_headers()
        elif self.path == '/index.html':
            content = PAGE.encode('utf-8')
            self.send_response(200)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(content))
            self.end_headers()
            self.wfile.write(content)
        elif self.path == '/stream.mjpg':
            self.send_response(200)
            self.send_header('Age', 0)
            self.send_header('Cache-Control', 'no-cache, private')
            self.send_header('Pragma', 'no-cache')
            self.send_header('Content-Type', 'multipart/x-mixed-replace;
boundary=FRAME')
            self.end_headers()
            try:
                while True:
                    with output.condition:
                        output.condition.wait()
                        frame = output.frame
                        self.wfile.write(b'--FRAME\r\n')
                        self.send_header('Content-Type', 'image/jpeg')
                        self.send_header('Content-Length', len(frame))
                        self.end_headers()

```

```

        self.wfile.write(frame)
        self.wfile.write(b'\r\n')
    except Exception as e:
        logging.warning(
            'Removed streaming client %s: %s',
            self.client_address, str(e))
    else:
        self.send_error(404)
        self.end_headers()

class StreamingServer(socketserver.ThreadingMixIn, server.HTTPServer):
    allow_reuse_address = True
    daemon_threads = True

with picamera.PiCamera(resolution='640x480', framerate=24) as camera:
    output = StreamingOutput()
    #Uncomment the next line to change your Pi's Camera rotation (in degrees)
    #camera.rotation = 90
    camera.start_recording(output, format='mjpeg')
    try:
        address = ('', 8000)
        server = StreamingServer(address, StreamingHandler)
        server.serve_forever()
    finally:
        camera.stop_recording()

```

Щоб зберегти файл, натисніть Ctrl + X, введіть Y і Enter.

Після написання скрипта можна запустити його за допомогою Python 3, використовуючи наступну команду:

```
python3 rpi_camera_surveillance_system.py
```

Після запуску сценарію можна отримати доступ до веб-серверу потокового відео за адресою:

http: // <Your_Pi_IP_Address>:8000.

Замініть своєю IP-адресою Raspberry Pi, в даному випадку `http://192.168.1.5:8000`. Можна також отримати доступ до потокового відео через будь-який пристрій, який має браузер і підключено до тієї ж мережі, що і RPi.

6.3 Розробка алгоритмів керування мобільного робота

Перед розробкою програми необхідно розробити алгоритм керування мобільного робота. Але перед самою розробкою алгоритму, необхідно визначити систему команд, які повинні обробляти та виконувати МР. Як вже було визначено, робот повинен рухатися вперед, назад, ліворуч, праворуч, а також робити зупинку двигунів. Отже, виходячи з цього, система команда наступна:

`forward` для руху вперед;

`backward` для руху назад;

`left` для руху ліворуч;

`right` для руху праворуч;

`stop` для зупинки двигунів.

Розробивши систему команд, перейдемо до розробки алгоритмів програми МР. Перший алгоритм, який необхідно розробити та представити – це алгоритм запуску усіх необхідних процесів для можливості керування мобільним роботом. На рисунку 6.6 приведена схема алгоритму завантаження процесів для керування. Цей алгоритм є лінійний без жодного розгалуження. На ньому лише представлено, у якій послідовності буде завантажено усі необхідні програмні компоненти.

На наступному етапі необхідно розробити схему алгоритму програми керування. Для більшої зручності, одну велику схему було розбито на декілька маленьких, а саме: узагальнену схему, схему обробки даних, та

схему виконання дій у випадку збігу команд з необхідними. На рисунку 6.7, приведено узагальнену схему алгоритму серверної програми керування МР. Нескінченний цикл необхідний для постійного прийому даних від клієнту та їх постійної обробки.

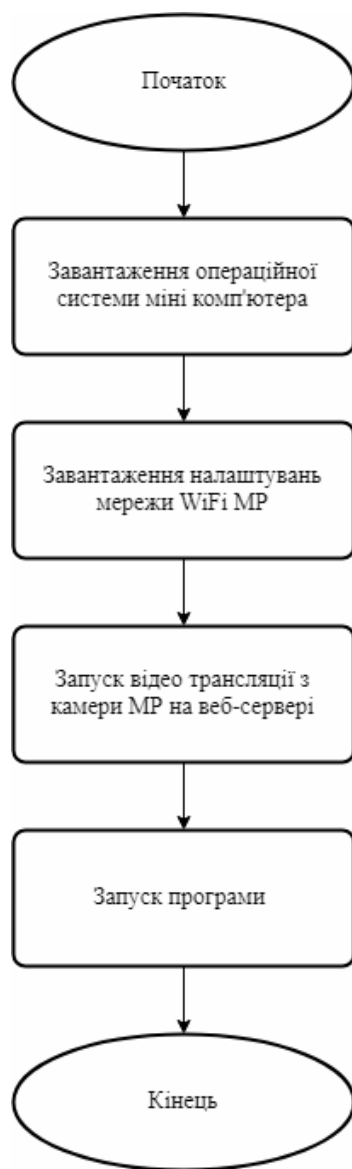


Рисунок 6.6 – Схема алгоритму завантаження процесів для керування



Рисунку 6.7 – Узагальнена схема алгоритму серверної програми керування MP

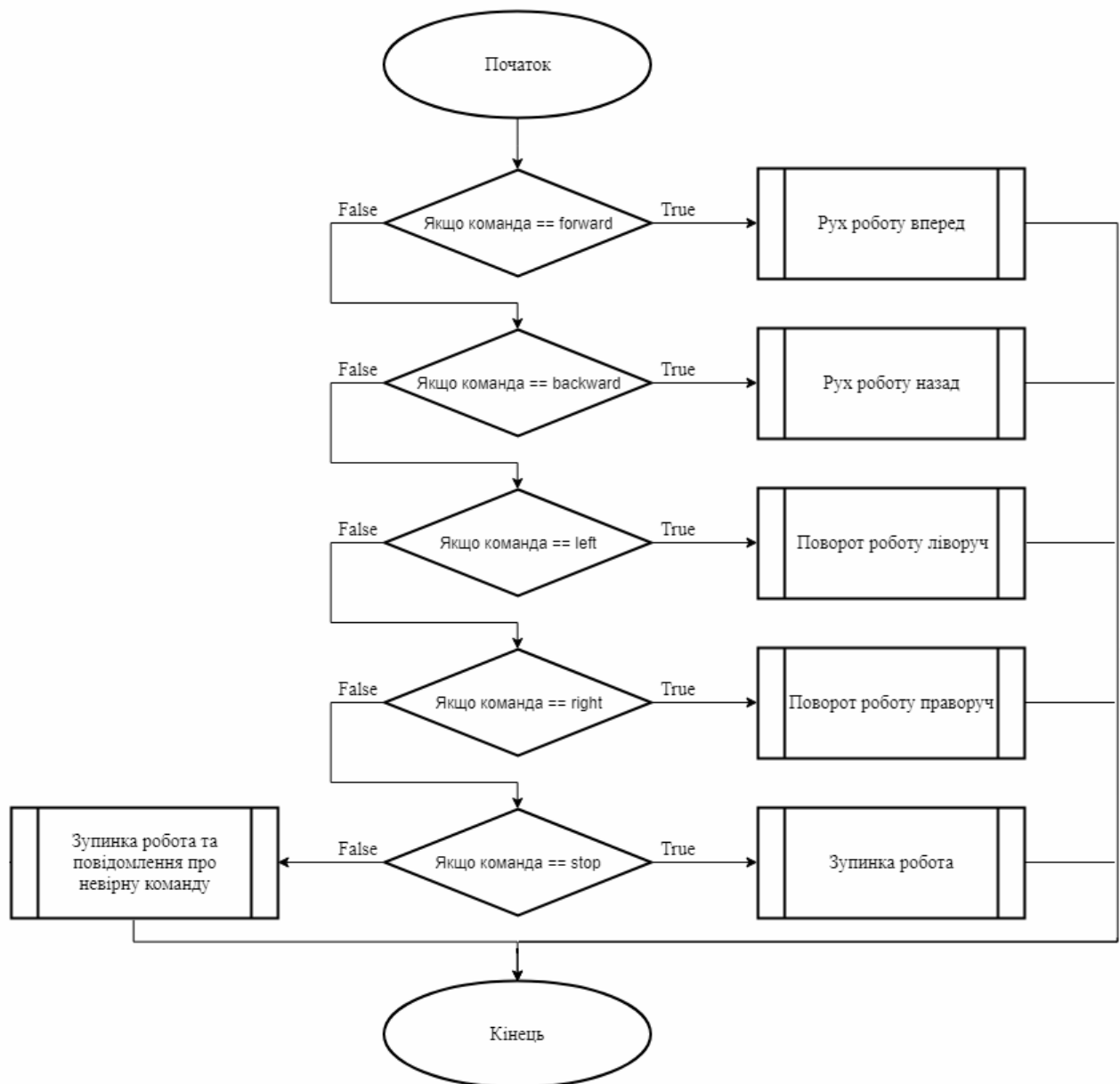


Рисунок 6.8 – Схема алгоритму прийому команд та їх обробки

Далі, слід розробити схему алгоритму обробки самих даних. Для реалізації цієї частини програми буде використано повне розгалуження (рис. 6.8). Також, в алгоритмі враховано випадок, якщо команда невідома.

Розробивши схеми великих ділянок програми, приступимо до маленьких ділянок програми, які будуть виконувати дії на самому роботі. На рисунку 6.9 приведено схему алгоритму ділянки програми, що виконує рух роботу вперед.

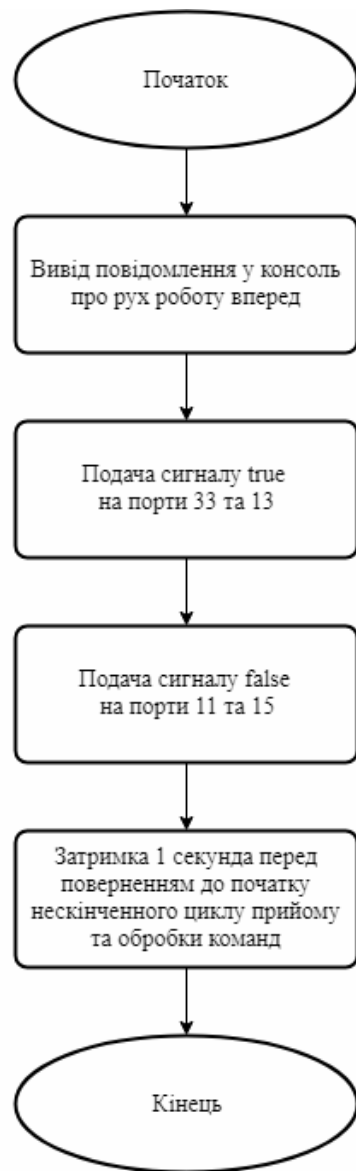


Рисунок 6.9 – Схема алгоритму підпрограми руху робота вперед

Як можна побачити, керування виконується завдяки подачі сигналів з плати керування, а саме міні комп'ютера RPi на драйвер двигунів. Таким чином, передається інструкція до цього самого драйверу, щодо полярності двигунів. Таким чином можна задати напрямок руху МР. На рисунку 6.10 приведена схема алгоритму руху МР назад.

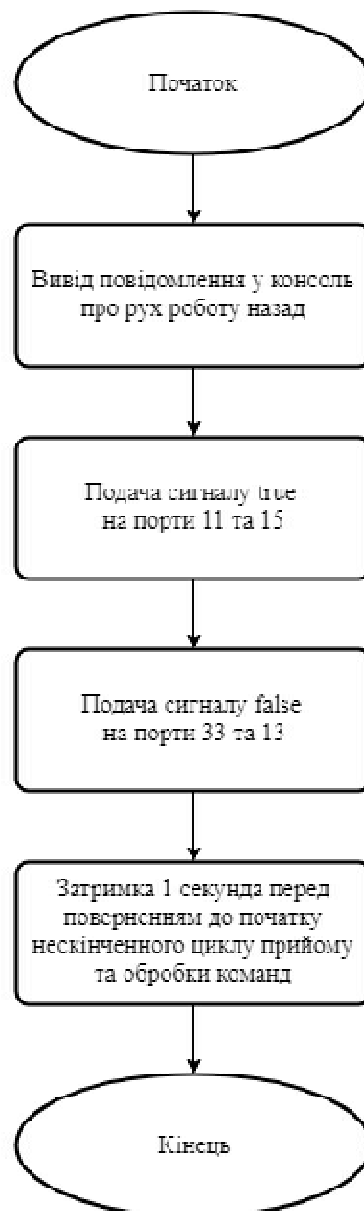


Рисунок 6.10 – Схема алгоритму підпрограми руху робота назад

Як видно зі схеми алгоритму (рис. 6.10), полярність двигунів змінюється, таким чином і змінюється напрямок руху МР на протилежний, на відміну від алгоритму (рис. 6.9), коли МР рухається вперед. На рисунку 6.11 приведена схема алгоритму повороту МР ліворуч.

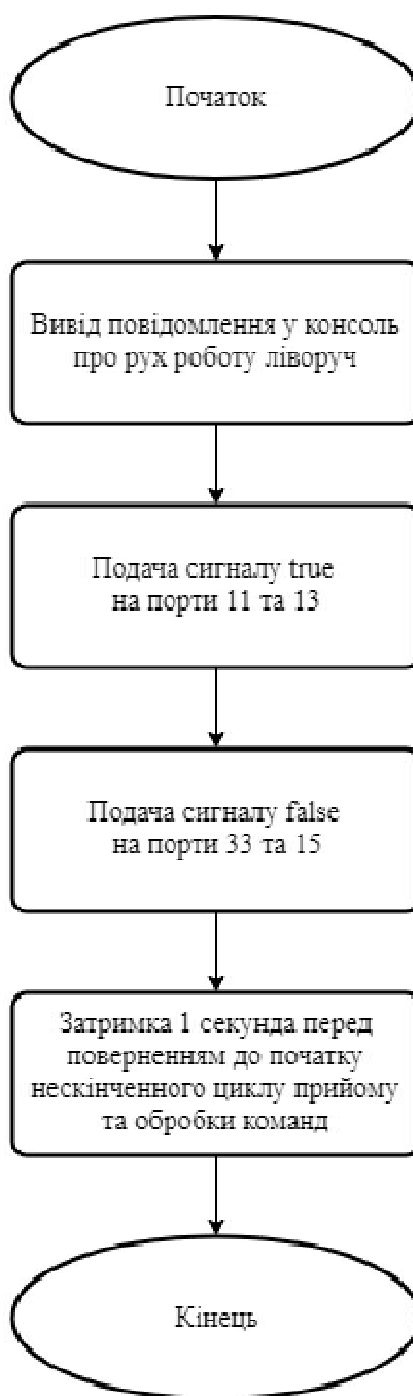


Рисунок 6.11 – Схема алгоритму підпрограми повороту робота ліворуч

Алгоритм руху МР ліворуч дозволяє робити поворот МР на місці. Це є дуже точний поворот, тому що робот виконує його без відхилень. Для цього встановлюється різна полярність на два двигуни, що є досить ефективним. На рисунку 6.12 приведено схему алгоритму повороту МР праворуч.

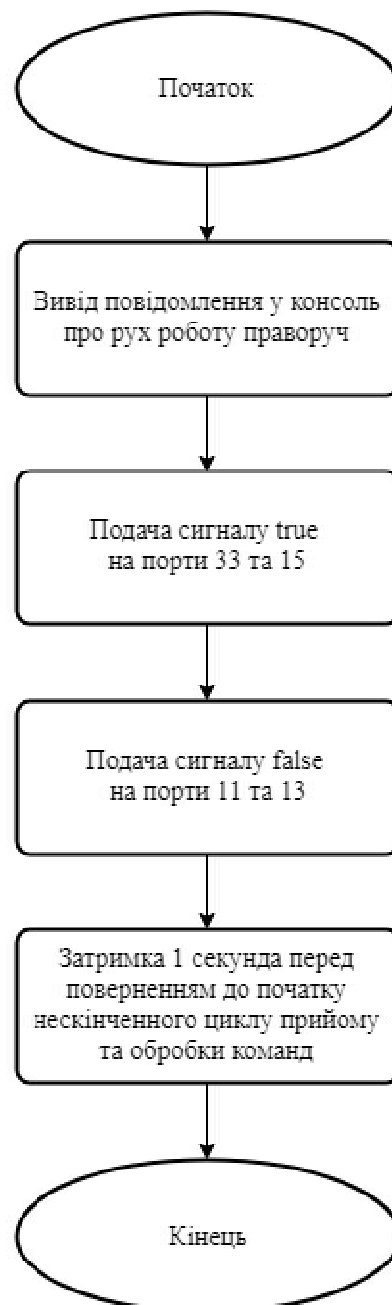


Рисунок 6.12 – Схема алгоритму підпрограми повороту робота праворуч

Так же як і рух ліворуч, поворот здійснюється на місці, але полярність двигунів протилежна, на відміну від повороту ліворуч. Також у кожній ділянці програми, яка виконує дії з роботом, присутня затримка 1 секунду, перед обробкою наступної команди. Це зроблено для запобігання утворення перешкод під час прийому великої кількості команд. На рисунку 6.13 приведено схему алгоритму зупинки роботи.

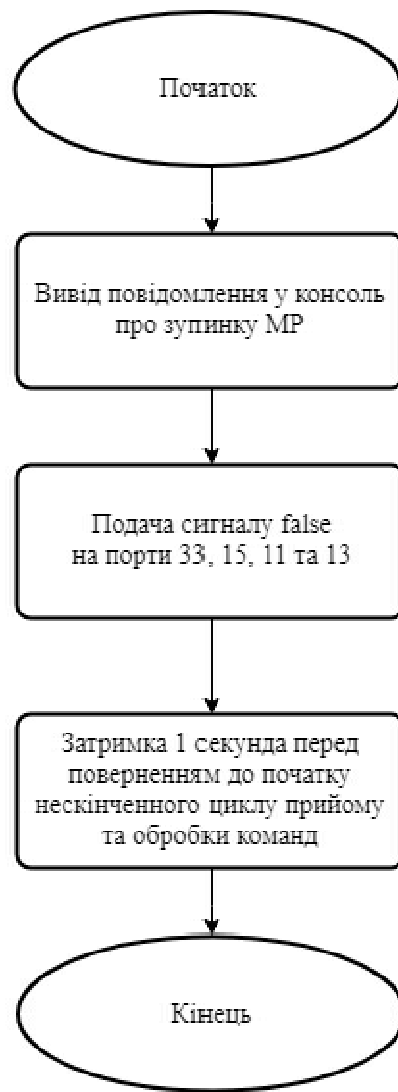


Рисунок 6.13 – Схема алгоритму підпрограми зупинки робота

Для зупинки робота, керуючий комп'ютер подає команду на драйвер двигунів вимкнути напругу. Таким чином, без напруги, робот негайно зупиняється. На рисунку 6.14, приведено схему алгоритму обробки невідомої команди.

Під час прийому невідомої команди роботом, комп'ютер одразу дає команду на драйвер відключити напругу на драйвер та виводить у термінал повідомлення про помилку.

Розробивши схему алгоритму програми робота, також необхідно розробити схему алгоритму роботи програми, яка передає команди на МР.

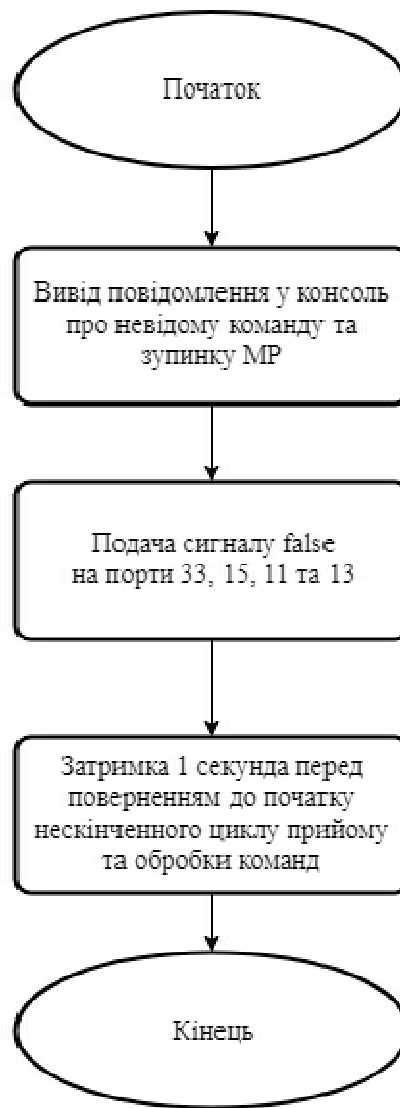


Рисунок 6.14 – Схема алгоритму підпрограми обробки невідомої команди

Програма повинна робити з'єднання з сокетом на сервері МР. Данні підключення потрібно вводити у форму, та після цього натиснути кнопку з'єднання. У випадку невдалого з'єднання необхідно відправити повідомлення про помилку. В іншому випадку, програма починає транслявати з веб-сервера відеозображення з камери МР за протоколом HyperText Transfer Protocol (HTTP) [342-345]. В інтерфейсі програми потрібно створити кнопки, які будуть відправляти необхідні команди на МР, а саме: Forward; Backward; Left; Right; Stop.

6.4 Розробка програми віддаленого керування мобільним роботом

Розробка програми робота є важливою частиною, тому що вона повинна приймати данні з клієнтської програми та обробляти їх. В якості мови програмування, буде використано мову Python 3, тому що вона дуже поширена у робототехніці, а тому знайти інформацію щодо використання потрібних функції не є складною задачею [346, 347]. Спочатку необхідно підключити бібліотеки, функції з яких будуть використані у програмі:

```
import RPi.GPIO as GPIO
```

```
import socket
```

```
import time
```

Бібліотека RPi.GPIO використовується для керування портами керуючого міні комп'ютера для зв'язку з драйвером двигунів. Бібліотека socket використовується для програмного створення каналу зв'язку між програмою-клієнтом та програмою-сервером [348]. Також бібліотека time використовується для створення затримки між операціями.

Наступним кроком необхідно налаштуємо режими роботи портів наступним кодом:

```
GPIO.setwarnings(False)
```

```
GPIO.setmode(GPIO.BOARD)
```

```
GPIO.setup(33, GPIO.OUT)
```

```
GPIO.setup(11, GPIO.OUT)
```

```
GPIO.setup(13, GPIO.OUT)
```

```
GPIO.setup(15, GPIO.OUT)
```

```
GPIO.setup(29, GPIO.OUT)
```

```
GPIO.setup(31, GPIO.OUT)
```

```
GPIO.output(29, True)
```

```
GPIO.output(31, True)
```

Таким чином було налаштовано режим звернення до портів та їх режими роботи. Наступним кроком необхідно відкрити сокет для обміну

даними:

```
UDP_IP = "0.0.0.0"
UDP_PORT = 5050
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.bind((UDP_IP, UDP_PORT))
```

Далі ініціалізуємо нескінченний цикл прийому та обробки даних. Дані на робота приходять у вигляді кортежу, тому необхідно записати команди, які потрібно обробляти також у вигляді кортежу, через неможливість порівняння строки із кортежем:

```
while True:
    raw=sock.recvfrom(1024)
    rawnew = ('forward', 'stop', 'backward', 'left', 'right')
```

Наступним кроком необхідно описати обробку прийнятого сигналу. В якості прикладу приведено умову, якщо прийнята команда на пересування роботу вперед. Якщо прийнята саме ця команда, робот рухається вперед, та робить затримку на 1 секунду перед обробкою наступної команди. Також у консоль виводиться налагоджувальна інформація у вигляді повідомлення, яка команда виконується:

```
if raw[0] == rawnew[0]:
    print 'robot moves forward'
    GPIO.output(33, True)
    GPIO.output(11, False)
    GPIO.output(13, True)
    GPIO.output(15, False)
    time.sleep(1)
```

Також, може статися, що робот прийняв невідому команду, тому слід обробити і її. Якщо робот прийняв команду, яка не задекларована у програмі, програма виведе повідомлення:

```
else:
    print "message error"
```

```

GPIO.output(33, False)
GPIO.output(11, False)
GPIO.output(13, False)
GPIO.output(15, False)
time.sleep(1)

```

6.5 Розробка клієнтської програми віддаленого керування мобільним роботом

Розробивши програму робота, яка приймає команди, також необхідно розробити програму, яка буде відправляти ці команди. В якості мови програмування було обрано мову C# [349] та .NET Framework 4.7 [350], тому що цей фреймворк та мова програмування дуже зручні, мають необхідні інструменти для роботи з віконними додатками. Також при розробці додатків на цій мові програмування дуже легко масштабувати розробляємий додаток. І ще, ця мова програмування має досить велике ком'юніті та авторитетного розробника мови, а саме, корпорацію Microsoft, яка у наш час дуже відома своїми продуктами, у тому числі Microsoft windows, Visual studio, Visual studio code та інші.

Почнемо розробку зі створення з'єднання по сокету з МР:

```

public Presenter(Form1 form, int port, string host){
    this.form = form;
    this.socket_port = port;
    socket = new Socket(AddressFamily.InterNetwork, SocketType.Dgram,
        ProtocolType.Udp);
    try{
        socket.Connect(host, port);
        this.form.ShowMessage("Connecting");
    }
    catch(Exception ex){

```

```

        this.form.ShowMessage(ex.ToString());
    }
}

```

В якості адреси з'єднання, програма бере дані з форми, у яку користувач повинен ввести адресу та порт підключення. Далі розробимо ділянку програми, яка оброблює подію натискання кнопки з'єднання. Після успішного з'єднання програми з МР, додаток одразу починає трансляцію з камери робота методом створення простого браузеру та об'єкту мови розмітки HyperText Markup Language (HTML)5, <iframe>, який створено саме для трансляції зі сторонніх веб-документів [351]:

```

private void bn_appy_configs_Click(object sender, EventArgs e){
    if (tb_port.Text.Length == 4 && tb_host.Text.Length >5) {
        presenter = new Presenter.Presenter(this,
            Convert.ToInt32(tb_port.Text), tb_host.Text );
        var page = @"
        <iframe src='http://'+ tb_host.Text + '
        width='100%' height='100%'> </iframe > ";
        webBrowser.DocumentText = page;
        webBrowser.ScriptErrorsSuppressed = true;
        MessageBox.Show("Configs has been saved");
    }else{
        MessageBox.Show("Uncorrect value");
    }
}

```

Також слід розробити у додатку обробник кнопок відправки команд на МР:

```

private void bn_KeyPress(object sender, KeyPressEventArgs e) {
    switch (((Button)sender).Name) {
        case "bn_command_1":
            presenter.Send_command("value",

```

```

        Commands.FORWARD);
        break;
    case "bn_command_2":
        presenter.Send_command("value",
            Commands.BACKWARD);
        break;
    case "bn_command_3":
        presenter.Send_command("value", Commands.LEFT);
        break;
    case "bn_command_4":
        presenter.Send_command("value", Commands.RIGHT);
        break;
    case "bn_command_5":
        presenter.Send_command("value", Commands.STOP);
        break;
    default:
        break;
}

```

Після створення обробника кнопок відправки команд на МР, також необхідно описати інтерфейс, який буде відправляти команди, залежно від натиснутої кнопки на МР:

```

public void Send_command(string value, Commands command) {
    if (socket != null){
        byte[] send_buf = new byte[200];
        switch (command){
            case Commands.FORWARD:
                send_buf = Encoding.ASCII.GetBytes("forward");
                break;
            case Commands.BACKWARD:

```

```

        send_buf
        Encoding.ASCII.GetBytes("backward");
        break;
    case Commands.LEFT:
        send_buf = Encoding.ASCII.GetBytes("left");
        break;
    case Commands.RIGHT:
        send_buf = Encoding.ASCII.GetBytes("right");
        break;
    case Commands.STOP:
        send_buf = Encoding.ASCII.GetBytes("stop");
        break;
    default:
        break;
}
socket.Send(send_buf);}}

```

6.6 Встановлення необхідних пакетів та налаштувань

Встановлення, та налаштування компонентів системи, є важливою частиною, щоб система керування працювала. Перше, що необхідно зробити, це встановити операційну систему, на базі якої буде працювати серверна частина МР. В якості операційної системи, потрібно встановити ОС Linux Raspbian 10 buster lite, яка є безкоштовною [352]. Наступним кроком, за допомогою програми Win32 Disk Imager потрібно розгорнути завантажений образ на флешці microSD. Вікно програми з усіма налаштуваннями приведено на рисунку 6.15.

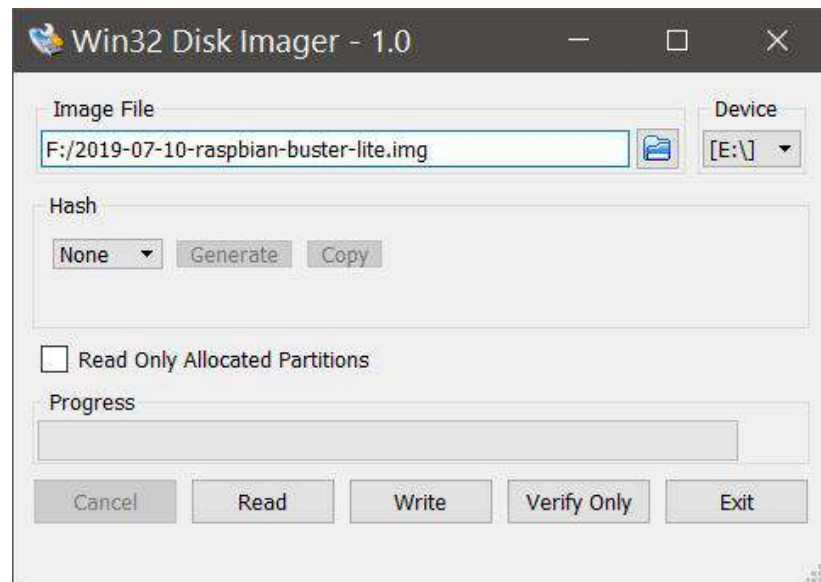


Рисунок 6.15 – Вікно програми Win32 Disk Imager

Після цього, необхідно встановити карту пам'яті у міні комп'ютер, подати напругу та з'єднатися з ним за протоколом SSH [354]. Стандартно, з'єднатися можна за адресою `raspberrypi`. Логін `pi`, пароль доступу `raspberrypi`. Перша команда, яку необхідно виконати, це `sudo su`, щоб отримати усі привілеї для виконання наступних операцій.

Наступним кроком, необхідно запустити утиліту *raspi-config*, обрати у розділі *Interfacing options* пункт *Camera* та включити інтерфейс. Вікно утиліти *raspi-config* приведено на рисунку 6.16.

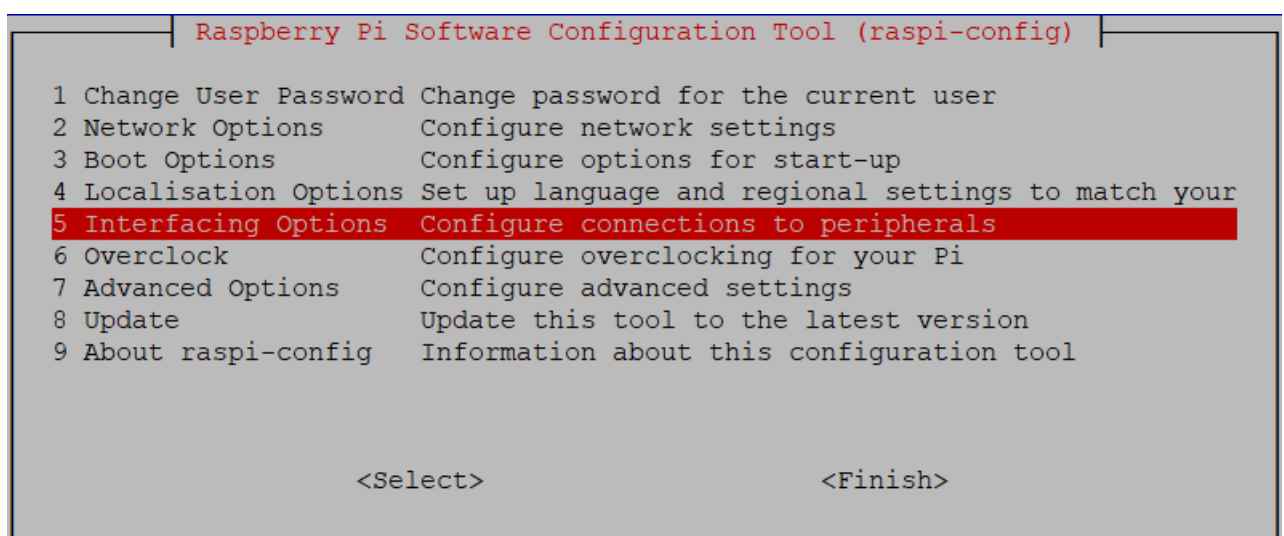


Рисунок 6.16 – Вікно утиліти raspi-config

Далі слід оновити список джерел для встановлення необхідних пакетів, виконавши команду *apt-get update*. Після цього встановимо усі необхідні пакети командою:

```
apt-get install hostapd dnsmasq git python-pip
```

За допомогою вже встановленої утиліти *pip*, встановимо необхідні бібліотеки Python для роботи програми робота:

```
pip install sockets times RPi.GPIO
```

Для встановлення утиліти трансляції відео встановимо пакет з репозиторія за допомогою утиліти *git* [20]:

```
git clone https://github.com/silvanmelchior/RPi_Cam_Web_Interface.git
```

Далі, слід перейти до директорії *RPi_Cam_Web_Interface* за допомогою команди *cd* та встановити утиліту:

```
cd /RPi_Cam_Web_Interface  
./install.sh
```

Після установки пакету, трансляція відео відбувається на веб сервері, та стартує автоматично, коли завантажується ОС.

Наступним кроком необхідно встановити програму керування. Для полегшення установки, програму завантажено на видалений репозиторій, тому встановлюється за допомогою утиліти *git*:

```
git clone https://github.com/aleksey-smx/robot_server_socket.git
```

Також, для можливості виконання програми, необхідно надати права на запуск командою:

```
chmod +x /home/pi/robot_server_socket/robot.py
```

Все необхідне зроблено, тепер потрібно зробити автоматичне виконання програми, під час завантаження ОС. Для цього необхідно додати у файл */etc/rc.local* наступну команду:

```
python /home/pi/robot_server_socket/robot.py
```

Останнє, що необхідно зробити, це налаштувати мережу WiFi, до якої повинен підключатись керуючий оператор. Для цього, необхідно відключити пакети:


```
systemctl stop dnsmasq
```

```
systemctl stop hostapd
```

Наступним кроком, необхідно додати у файл `/etc/dhcpd.conf`:

```
interface wlan0
```

```
static ip_address=192.168.7.1/24
```

```
nohook wpa_supplicant
```

Далі слід перезапустити пакет `dhcpd`:

```
service dhcpd restart
```

Також потрібно відредагувати файл `/etc/dnsmasq.conf`, де задамо діапазон адрес клієнтів:

```
interface=wlan0
```

```
dhcp-range=192.168.7.2,192.168.8.20,255.255.255.0,24h
```

Налаштуємо параметри бездротової мережі у файлі `/etc/hostapd/hostapd.conf`:

```
interface=wlan0
```

```
driver=nl80211
```

```
ssid=Назва мережі
```

```
hw_mode=g
```

```
channel=7
```

```
wmm_enabled=0
```

```
macaddr_acl=0
```

```
auth_algs=1
```

```
ignore_broadcast_ssid=0
```

```
wpa=2
```

```
wpa_passphrase=пароль
```

```
wpa_key_mgmt=WPA-PSK
```

```
wpa_pairwise=TKIP
```

```
rsn_pairwise=CCMP
```

У файлі */etc/default/hostapd* додаємо завантаження конфігурації бездротової мережі:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

Запустимо пакет *hostapd*:

```
systemctl unmask hostapd
```

```
systemctl enable hostapd
```

```
systemctl start hostapd
```

Додамо у файл */etc/sysctl.conf* наступний параметр:

```
net.ipv4.ip_forward=1
```

А також створимо правило для фаєрвола та збережемо його у пам'яті:

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

```
sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
```

Також додамо у файл */etc/rc.local* автоматичне встановлення правил при перезавантаженні:

```
iptables-restore < /etc/iptables.ipv4.nat
```

На цьому, встановлення та налаштування програмного забезпечення завершується. Достатньо перезавантажити міні комп'ютер, щоб усі параметри вступили в силу.

7 РЕАЛІЗАЦІЯ СИСТЕМИ ІДЕНТИФІКАЦІЇ ОБ'ЄКТІВ МЕТОДОМ YOLO МООВОЮ РАУТЕН

7.1 Класифікація задач розпізнавання образів

Сьогодні інформаційні системи розпізнавання образів (РО) є складовою частиною будь-яких штучних інтелектуальних систем і використовуються в експертних системах, базах даних та базах знань, в інформаційних, прогнозних та інших системах. Прикладами можуть бути комплексні інтелектуальні системи наприклад, «Здобуття знань» («DataMining») для прийняття рішень, які використовують ряд методів РО. На рисунку 7.1 наведено систему пошуку та спосіб здобуття знань [354-356].

Будь-які автоматичні чи автоматизовані інтелектуальні системи, включно з РО, повинні мати однакові риси («мету», «об'єкт зовнішнього середовища», «розпізнавання об'єкта за ознаками») та повинні виконувати відповідні «оптимальні інтелектуальні дії».

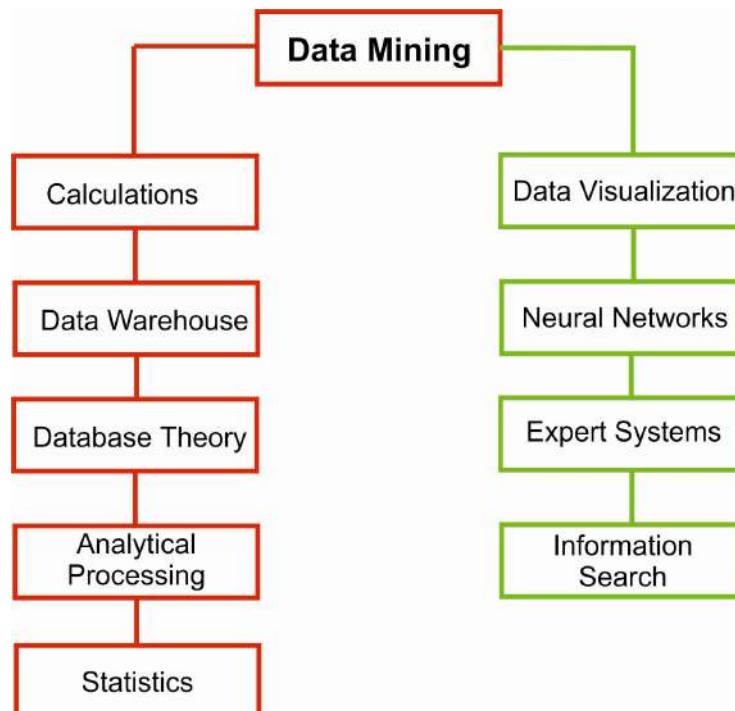


Рисунок 7.1 – Система «Здобуття знань»

Але різниця між «Розпізнаванням образів» та згаданими вище напрямками все ж є: дисципліна «Розпізнавання образів» узагальнює, об'єднує, досліджує і удосконалює основні оптимальні методи, алгоритми, моделі і засоби інтелектуальних систем при обробці даних реального чи абстрактного середовища, в той час як інші згадані інтелектуальні системи повинні використовувати і поглиблювати ці загальні методи дисципліни РО з урахуванням особливостей процесів у конкретних напрямках і ситуаціях.

Система рівнів розпізнавання образу ґрунтується на характеристичних ознаках, які використовуються у процесі ініціалізації. Можлива класифікація систем розпізнавання показана на рисунку 7.2.

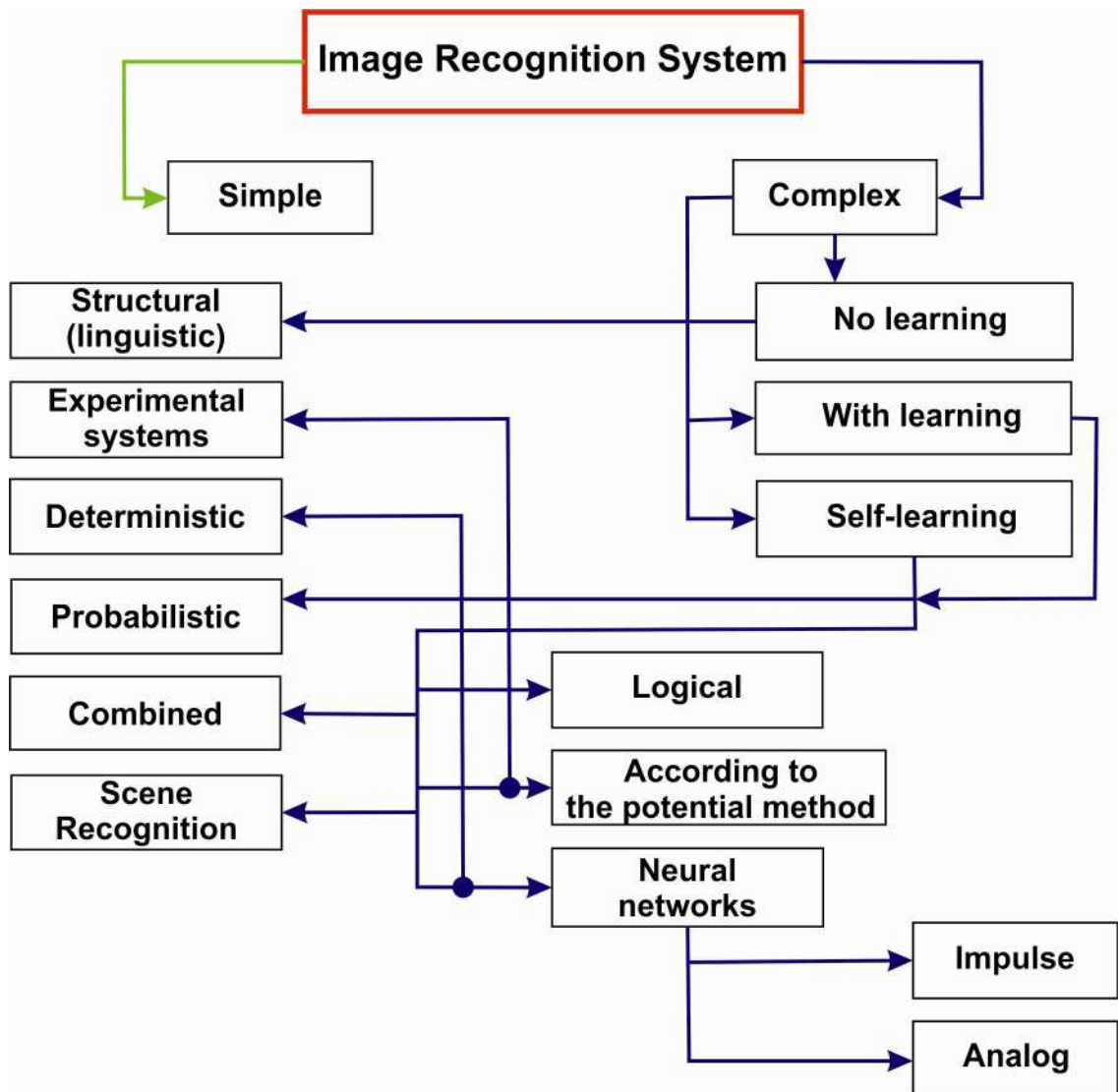


Рисунок 7.2 – Розподіл системи розпізнавання

Самонавчальна система використовує набір правил, за якими сама виробляє класифікацію, яка може відрізнятися від природної.

Приклад застосування експертних систем – існування евристичних або інтуїтивних методів прийняття рішень і відсутність точних алгоритмів або розрахунків. Проста експертна система містить накопичені знання багатьох експертів у певній предметній області і здатна в рамках цієї області класифікувати об'єкти, давати відповіді, рекомендації, поради, запитуючи при необхідності додаткову інформацію. Експертна система найчастіше використовує правила, які мають форму: «*if, then, else*» [357-359]. До експертних систем можна віднести медичні системи (по ознаках розпізнають хвороби), ситуаційні системи (розпізнають ситуацію і приймають рішення стосовно наступних дій).

Детерміновані системи. У цих системах для побудови алгоритмів розпізнавання використовують методи, засновані на вимірюванні відстаней між розпізнаваним об'єктом і еталонами класів.

Імовірнісні системи використовують імовірнісні методи розпізнавання, засновані на теорії статистичних рішень.

Логічні системи використовують логічні методи розпізнавання, засновані на наявності логічних зв'язків, в яких змінні – логічні ознаки розпізнаваних об'єктів, а невідомі величини – класи, до яких ці об'єкти відносяться.

Системи РО з використанням метода потенціалів, за яким ознака об'єкту розглядається як його електричний потенціал, який зменшується із зростанням відстані до об'єкту.

Нейромережні систем використовують моделі нейронів для розпізнавання образів і розділяються на імпульсні та аналогові системи: імпульсні складаються з нейронів, що отримують на входи кортежі імпульсів (спайків) та генерують на виходах імпульси; аналогові складаються з нейронів, які аналізують безперервний сигнал.

Комбіновані системи РО використовують комбінації методів та систем. Приклади комбінованих систем – інтелектуальний аналізатор даних DataMining.

7.2 Основні поняття побудови алгоритмів на базі CNN

CNN – мережі, що базуються на принципах роботи візуальної кори людського мозку [360-362].

Згортова нейронна мережа (ConvNet/CNN) – це алгоритм глибокого навчання, який може приймати вхідне зображення, привласнювати важливість аспектам або об'єктам зображенні та відрізняти одне від одного [363-365]. При цьому зображення порівняно з іншими алгоритмами вимагають набагато менше попередньої обробки. У примітивних методах фільтри розробляються вручну, але достатньо навчені мережі CNN вчаться застосовувати ці фільтри / характеристики.

Архітектура CNN аналогічна структурі зв'язків нейронів у мозку людини, вчені черпали натхнення в організації зорової кори головного мозку. Окремі нейрони реагують на стимули лише у деякій області поля зору, також відомого як перцептивне поле. Багато перцептивних полів перекриваються, повністю покриваючи поле зору CNN. Приклад архітектури CNN наведено на рисунку 7.3.

Мережа CNN здатна з успіхом охоплювати просторові та часові залежності у зображенні через застосування відповідних фільтрів. Така архітектура за рахунок скорочення кількості задіяних параметрів та можливості повторного використання ваги дає кращу відповідність набору даних зображень. Іншими словами, мережу можна навчити краще розуміти складність зображення.

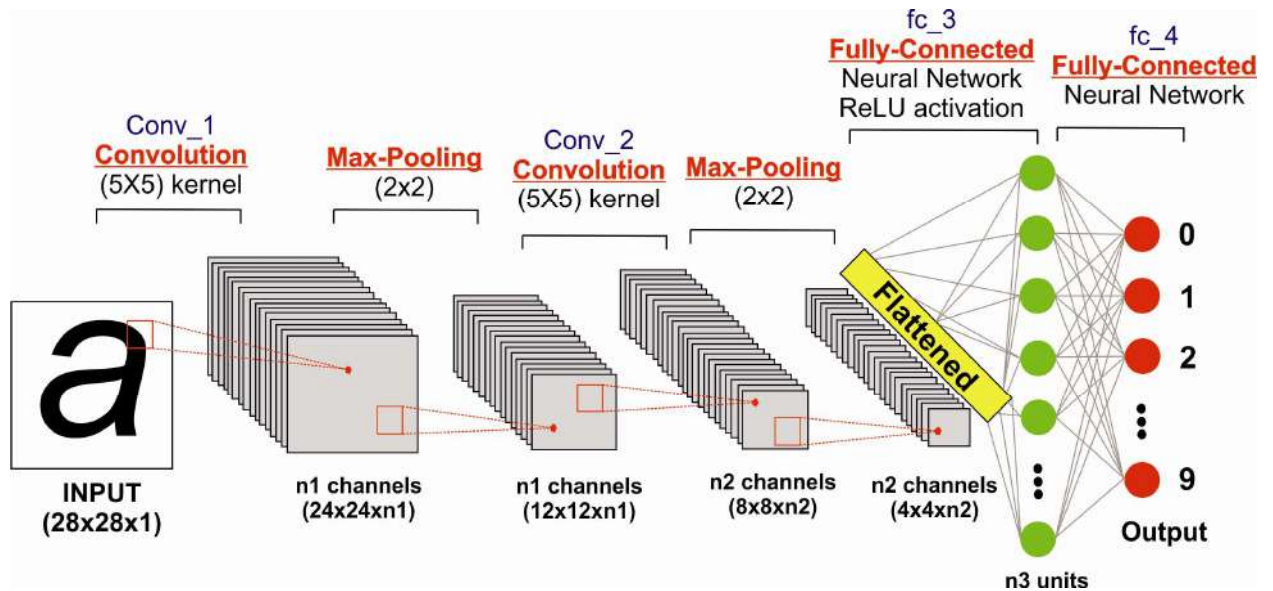
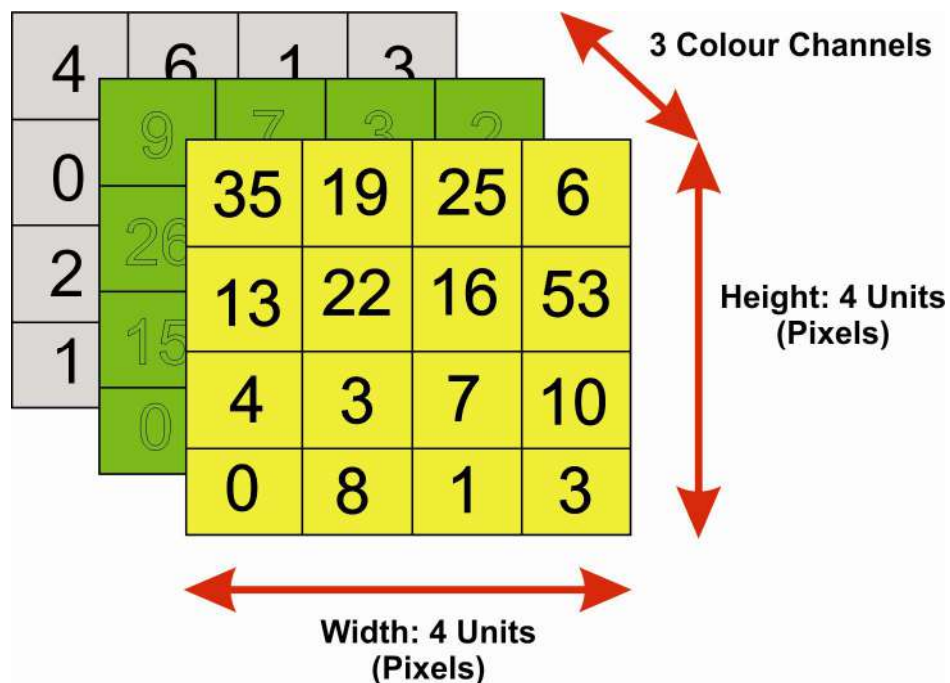


Рисунок 7.3 – Архітектура CNN

На рисунку 7.4 видно розділене на три колірні площини (червону, зелену та синю) RGB-зображення, яке можна описати в різних колірних просторах – у відтінках сірого (Grayscale), RGB [366], HSV [367], CMYK [368] і т.д.

Рисунок 7.4 - RGB-зображення $4 \times 4 \times 3$.

Можна уявити, наскільки інтенсивними будуть обчислення, коли зображення досягнуть розмірів, наприклад, 8 К (76804320). Роль CNN

полягає в тому, щоб привести зображення у форму, яку легко обробляти, без втрати ознак, що мають вирішальне значення в отриманні позитивного прогнозу. Це важливо при розробці архітектури, яка не тільки добре вивчає функції, але масштабується для масивних наборів даних. Графічний приклад згортки зображення $5 \times 5 \times 1$ з ядром $3 \times 3 \times 1$ для отримання згорнутої ознаки $3 \times 3 \times 1$ представлений на рисунку 7.5.

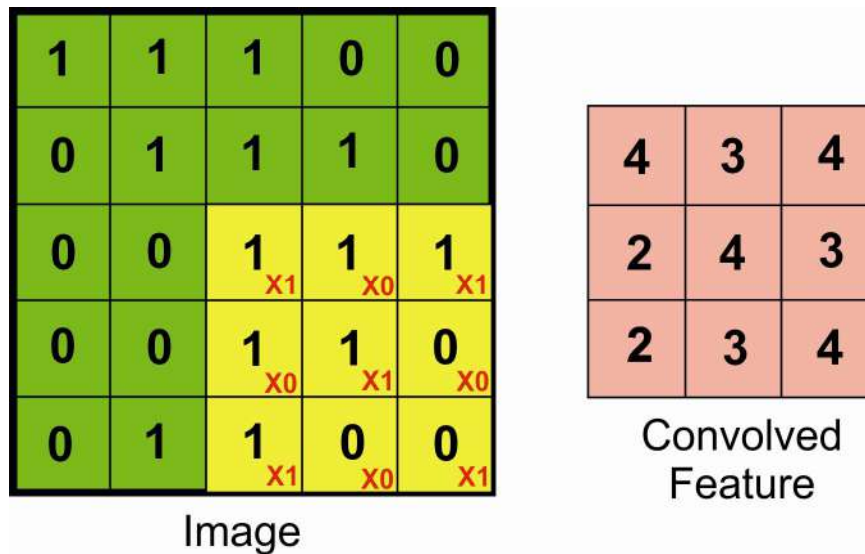


Рисунок 7.5- Згортка зображення $5 \times 5 \times 1$ з ядром $3 \times 3 \times 1$ для отримання згорнутої ознаки $3 \times 3 \times 1$

На рисунку 7.5 зелена секція нагадує наше вхідне зображення $5 \times 5 \times 1$. Елемент, який бере участь у виконанні операції згортки у першій частині шару згортки, називається ядром / фільтром K , він представлений жовтим кольором. Нехай K буде матрицею $3 \times 3 \times 1$:

$$\text{Kernel / Filter} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \quad (7.1)$$

Ядро зміщується 9 разів через довжину кроку в одиницю (тобто кроку немає), щоразу виконуючи операцію множення матриці K на матрицю P , над якою знаходиться ядро. Приклад переміщення ядра представлений рис. 7.6.

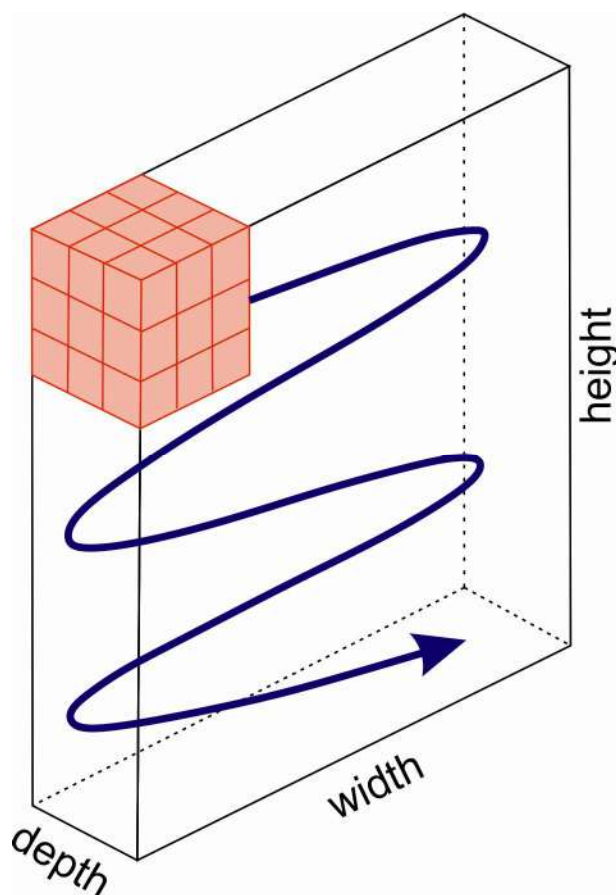


Рисунок 7.6 – Переміщення ядра

Фільтр переміщається вправо з певним значенням кроку, доки не проаналізує всю ширину. Рухаючись далі, він переходить до початку зображення (ліворуч) з тим самим значенням кроку і повторює процес доти, доки проходить все зображення. Приклад операцій згортки на матриці зображення $M \times N \times 3$ з ядром $3 \times 3 \times 3$ представлений на рисунку 7.7.

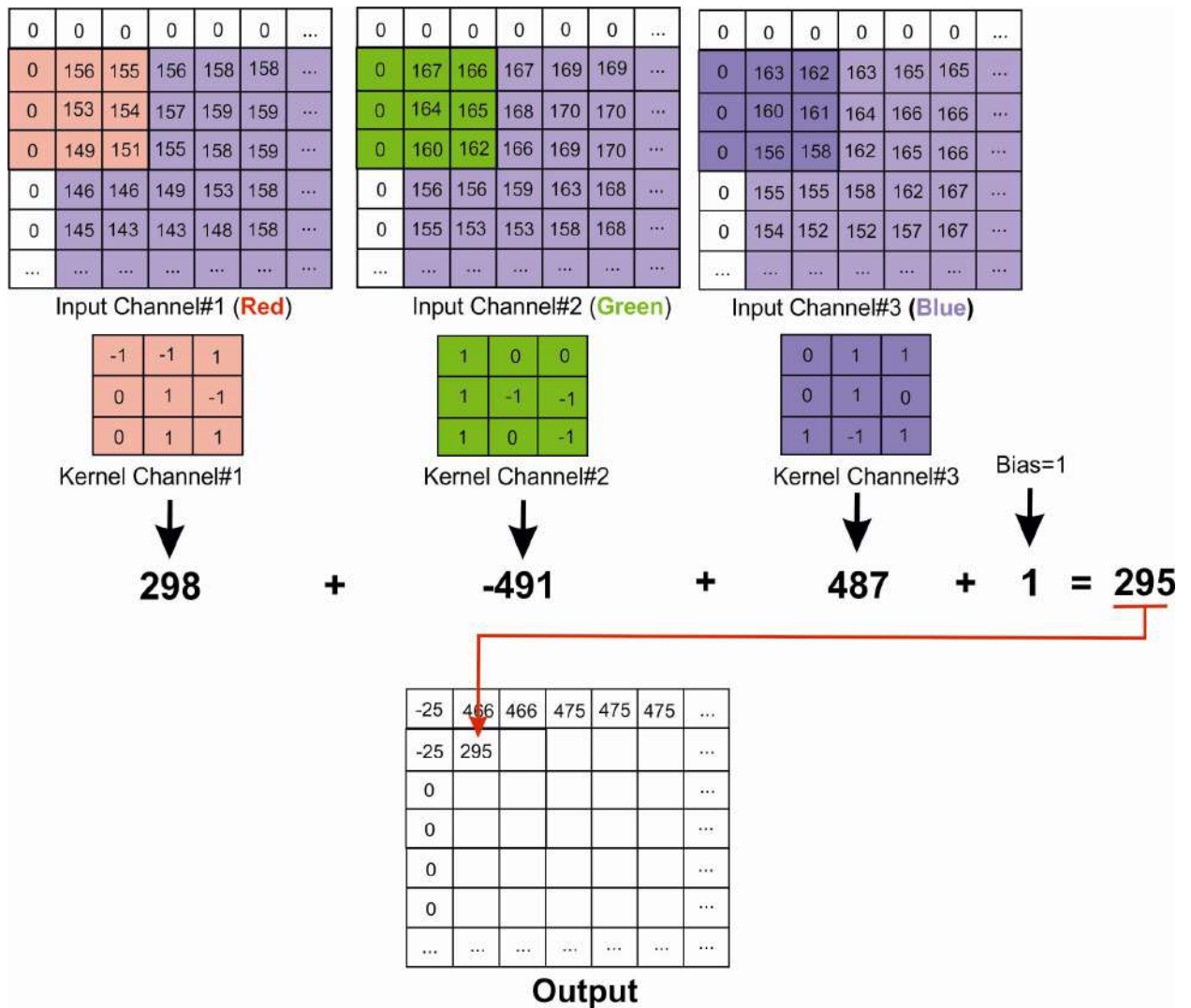


Рисунок 7.7 – Операція згортки на матриці зображення $M \times N \times 3$ з ядром $3 \times 3 \times 3$

Якщо розглядати зображення з кількома каналами (наприклад, RGB) ядро має таку саму глибину, що й у вхідного зображення. Матричне множення виконується між стеками K_n і I_n ($[K_1, I_1]; [K_2, I_2]; [K_3, I_3]$), всі результати підсумовуються зі зсувом, щоб отримати ууцільнений канал виведення згорнутих ознак з глибиною в 1. Операція згортки з довжиною кроку, що дорівнює 2, наведено на рисунку 7.8.

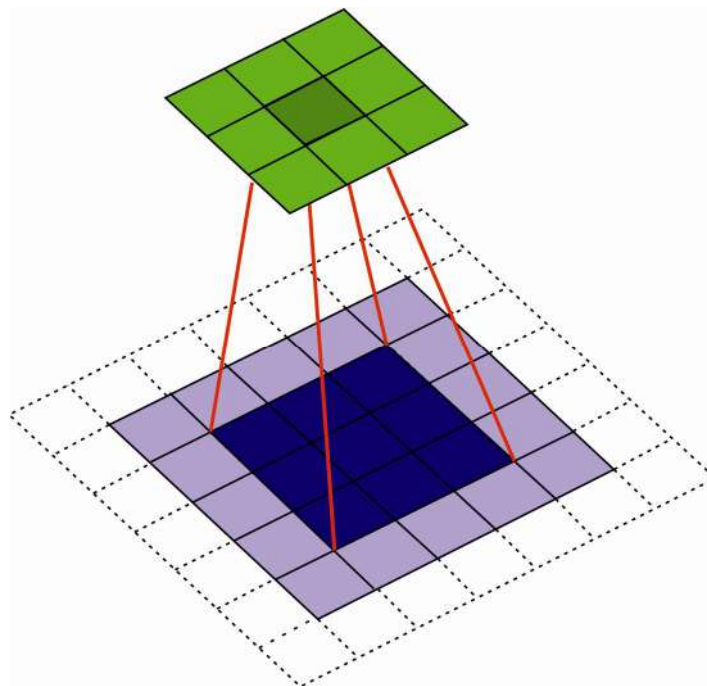


Рисунок 7.8 – Операція згортки з довжиною кроку 2

Згортка робиться, щоб витягти високорівневі ознаки, наприклад, краї вхідного зображення. Сітку не потрібно обмежувати єдиним шаром. Перший шар умовно відповідає за захоплення ознак низького рівня, таких як кромки, колір, орієнтація градієнта і т. д. Через додаткові шари архітектура адаптується до ознак високого рівня, ми отримуємо сітку з розумінням зображень у наборі даних, схожому на наше.

Результатів згортки мають два типи: перший – згорнута ознака зменшується у розмірі порівняно з розміром на вході, другий тип стосується розмірності – вона залишається колишньою, або збільшується. Це робиться шляхом застосування допустимого заповнення у першому випадку, або нульового заповнення – у другому. Приклад нульового заповнення наведено рисунку 7.9.

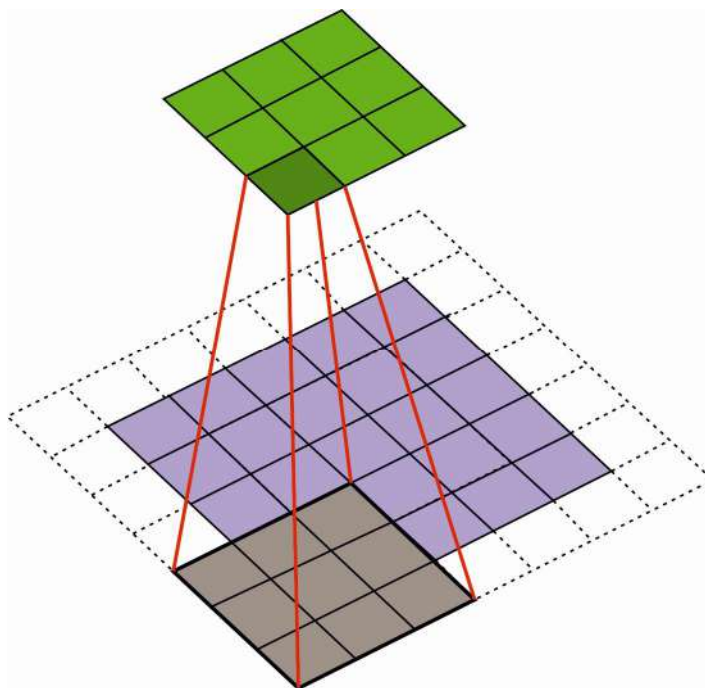


Рисунок 7.9 – Нульове заповнення: для створення зображення $6 \times 6 \times 1$ зображення $5 \times 5 \times 1$ доповнюється нулями

Збільшуючи зображення $5 \times 5 \times 1$ до $6 \times 6 \times 1$, а потім проходячи над ним ядром $3 \times 3 \times 1$, виявляється, що згорнута матриця матиме роздільну здатність $5 \times 5 \times 1$. Звідси і назва – нульове заповнення. З іншого боку, проробивши те саме без заповнення, отримаємо матрицю з розмірами самого ядра ($3 \times 3 \times 1$); ця операція називається допустимим наповненням.

Подібно до згорткового шару, шар об'єднання відповідає за зменшення розміру згорнутого об'єкта в просторі. Це робиться задля зменшення необхідної при обробці даних обчислювальної потужності, за рахунок скорочення розмірності. Крім того, це корисно для отримання домінуючих ознак, які є обертальними і позиційними інваріантами, тим самим дозволяючи підтримувати процес ефективного навчання моделі.

Є два типи об'єднання: максимальне та середнє. Перше повертає максимальне значення із покритої ядром частини зображення. А середнє об'єднання повертає середнє значення зі всіх значень покритої ядром частини. Приклад типів з'єднання наведено на рис. 7.10.

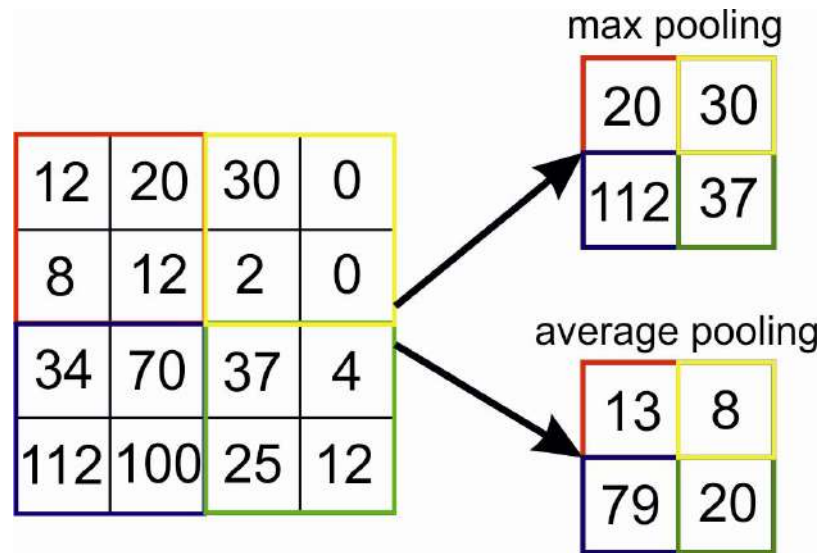


Рисунок 7.10 – Типи об'єднання

Максимальне об'єднання також виконує функцію шумоподавлення. Воно повністю відкидає зашумлені активації, а також усуває шум разом із зменшенням розмірності. З іншого боку, середнє поєднання для придушення шуму просто знижує розмірність. Отже, можна сказати, що максимальне об'єднання працює набагато краще за середнє.

Шари об'єднання та згортки разом утворюють i -тий шар згорткової нейронної мережі. Кількість таких шарів може бути збільшено залежно від складності зображень, щоб краще схоплювати деталі, але це робиться за рахунок збільшення обчислювальної потужності. Виконання процесу описаного вище дозволяє моделі розуміти особливості зображення. Перетворимо результат в стовпцевий вектор і надамо його звичайній нейронній мережі для класифікації.

Додавання повнозв'язкового шару – це (зазвичай) обчислювально недорогий спосіб навчання нелінійним комбінаціям високорівневих ознак, які представлені на виході шару згортки. Повнозв'язний шар вивчає функцію у цьому просторі, яка може бути нелінійною.

Після перетворення вхідного зображення в придатну для багаторівневого перцептрону форму необхідно згладити зображення у вектор стовпець. Згладжений вихідний сигнал подається на нейронну мережу з

прямим зв'язком, при цьому на кожній ітерації навчання застосовується зворотне поширення. За серію епох модель знаходить здатність розрізняти домінуючі та деякі низькорівневі ознаки у зображеннях та класифікувати їх методом класифікації Softmax.

У CNN є різні архітектури, що зіграли ключову роль у побудові алгоритмів, на яких стоїть і в найближчому майбутньому стоятиме штучний інтелект в цілому. Деякі з цих архітектур перераховані нижче:

- LeNet [369];
- AlexNet [370];
- VGGNet [371];
- GoogLeNet [372];
- ResNet [373];
- ZFNet [374].

7.3 Програмна реалізація систем ідентифікації об'єктів методом YOLO

Сучасні системи ідентифікації змінюють призначення класифікаторів для виконання виявлення. Щоб виявити об'єкт, ці системи беруть класифікатор для цього об'єкта та оцінюють його в різних місцях і масштабують на тестовому зображенні. Такі системи, як деформовані деталі моделі (DPM) використовують підхід з розсувним вікном [375], де класифікатор виконується в рівномірно розташованих місцях по всьому зображенню (рис. 7.11).

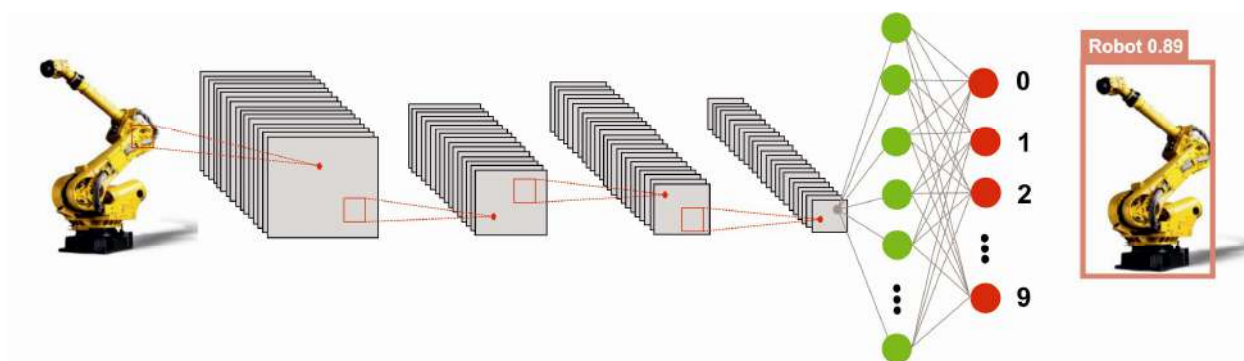


Рисунок 7.11 – Приклад системи виявлення YOLO

Наша система змінює розміри вхідного зображення до 448×448 , потім запускає одну згорткову мережу на зображенні і обмежує результати виявлення за допомогою моделі. Одна згортка мережі одночасно прогнозує кілька обмежувальних прямокутників і ймовірності класів для цих блоків. Тренування відбувається на повних зображеннях і безпосередньо оптимізується продуктивність виявлення. Ця уніфікована модель має ряд переваг над традиційними методами виявлення об'єктів.

По-перше, YOLO надзвичайно швидкий. Оскільки визначаємо кадр як проблему регресії, тому не потрібен складний конвеєр.

По-друге, YOLO мислить глобально про зображення, коли робити прогнози. На відміну від розсувного вікна та методів заснованих на пропозиції, YOLO бачить ціле зображення під час навчання та тестування, тому він неявно кодує контекстну інформацію про класи, а також їх зовнішній вигляд.

По-третє, YOLO вивчає узагальнювані уявлення об'єктів. Під час навчання на природних зображеннях та тестування на ілюстраціях YOLO перевершує найкращі методи виявлення, як-от DPM і R-CNN з великим відривом. Оскільки YOLO дуже узагальнений, він має меншу ймовірність руйнування при застосуванні нових доменів або неочікуваних вхідних даних.

Мережа використовує функції з усього зображення, щоб передбачити кожну обмежувальну рамку. Це також прогнозує всі обмежувальні рамки для всіх класів для зображення одночасно. Це означає, що наша мережа

глобально міркує про повне зображення та всі об'єкти на зображенні. Конструкція YOLO забезпечує наскрізне навчання та швидкість в реальному часі, зберігаючи високу середню точність [376].

Можемо сказати, що система розбиває вхідне зображення на сітку $S \times K$. Якщо центр об'єкта потрапляє в клітинку сітки, ця клітинка сітки відповідає за виявлення цього об'єкта. Даний спосіб продемонстровано на рисунку 7.12.

Кожна обмежувальна рамка складається з 5 передбачень: x , y , w , h , і так звана довіра. Координати (x, y) представляють центр квадрата відносно меж клітинки сітки. Ширина і висота прогножуються відносно всього зображення. Прогноз довіри представляє собою IOU між прогнозованим боксом і будь-яким [377].

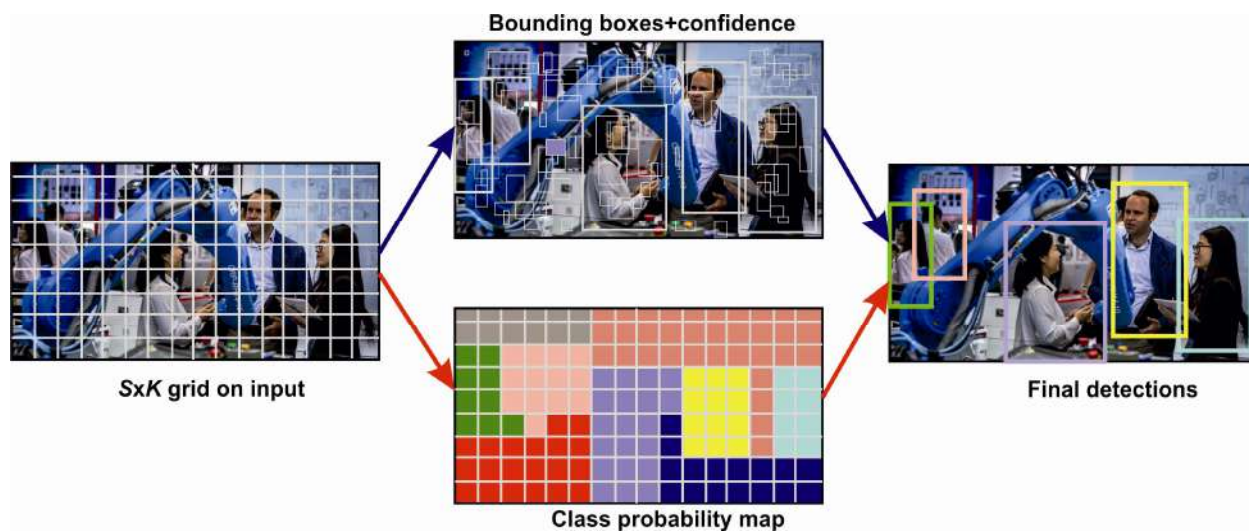


Рисунок 7.12 – Поділ зображення на сітку $S \times S$ і для кожної клітинки сітки передбачає «В» обмежувальні рамки

Мережева архітектура приймає GoogLeNet модель для класифікації зображень. Є 24 згорткові шари, за якими слідує 2 повністю пов'язаних шари.

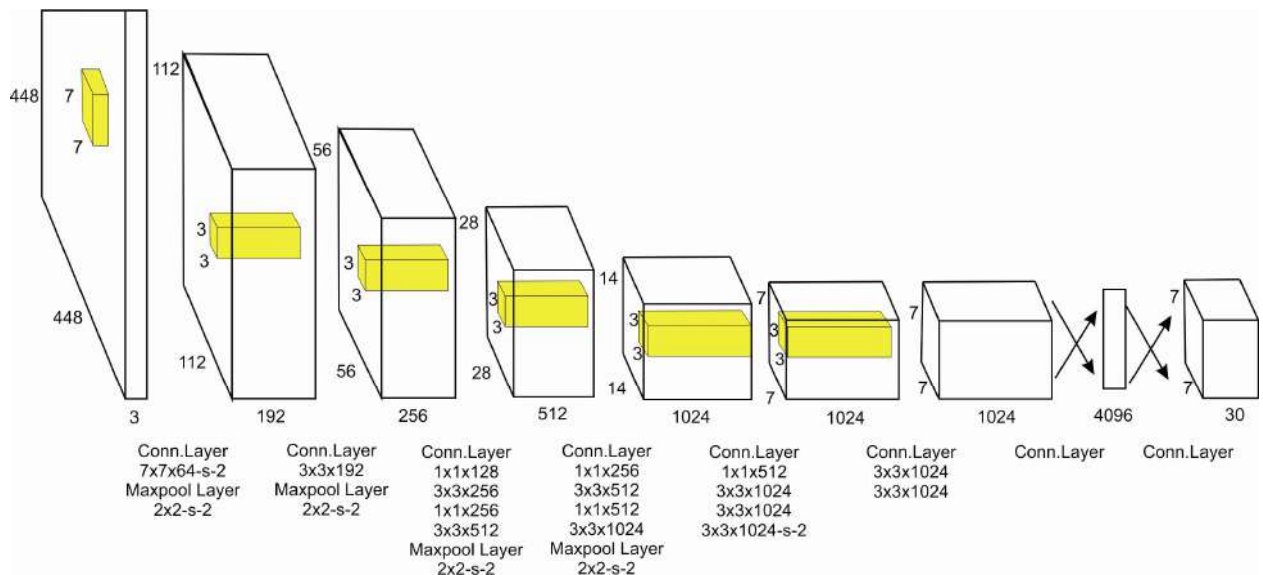


Рисунок 7.13 – Архітектура нейронної мережі, принцип навчання та виявлення об'єкта

Даний метод використовує функцію лінійної активації для кінцевого шару і всі інші шари використовують наступну лінійну активацію (рис. 7.13):

$$\varphi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \quad (7.2)$$

Відбувається оптимізація похибки суми-квадратів у результатах досліджуваної моделі за (7.1).

YOLO передбачає декілька обмежувальних прямокутників на клітинку сітки. Під час навчання потрібен лише один обмежувальний прямокутник, котрий буде відповідати за кожен об'єкт. Призначаємо один прямокутник бути «відповідальним» за передбачення об'єкта, на основі якого прогноз має найвищий поточний IOU. Це призводить до спеціалізації між обмежувальною рамкою прямокутника. Кожен прямокутник стає кращим у передбаченні певного розміру, співвідношення сторін або класи об'єктів, покращуючи в цілому узгодженість.

Виявлення об'єктів є основною проблемою комп'ютерного зору. Даний розділ передбачає порівняння YOLO з кількома найкращими системами виявлення, висвітлюючи ключові схожості та відмінності:

– Deformable parts models (DPM) – моделі деталей, що деформуються. Даний спосіб використовує підхід ковзного вікна для виявлення об'єктів (рис. 7.14);

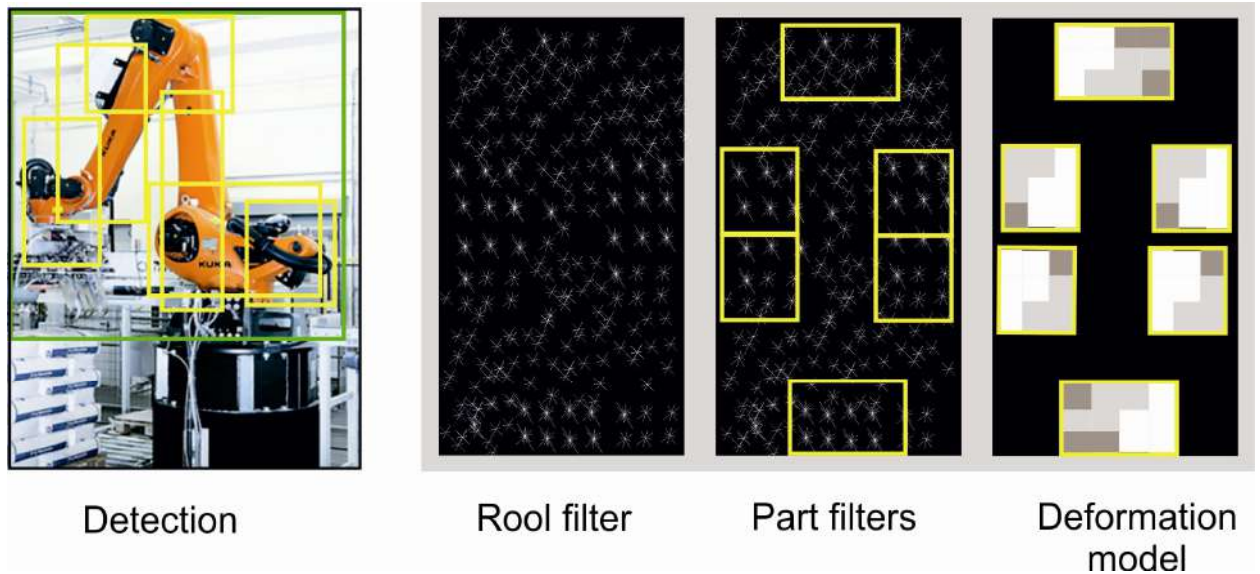


Рисунок 7.14 – Приклад розбиття зображення на рівні (DPM)

DPM використовує роз'єднаний конвеєр для вилучення статичних функцій, класифікувати регіони, передбачити обмежувальні рамки для високої оцінки регіону тощо. Наша система замінює всі ці розрізнені частини з єдиною згортковою нейронною мережею. Мережа одночасно виконує виділення ознак, передбачення обмежувальної рамки, немаксимальне придушення та контекстне міркування. Замість статичних функцій мережа тренує функції і оптимізує їх для завдання виявлення. Наявність уніфікованої архітектури веде до швидшої та більш точної моделі ніж DPM;

– R-CNN. Модель даного типу використовує пропозиції регіонів замість розсувних вікон для пошуку об'єктів на зображеннях. Вибірковий пошук генерує потенційні обмежувальні рамки, згортка мережі витягує ознаки, SVM оцінює рамки, лінійна модель налаштовує обмежувальні рамки,

а немаксимальне придушення усуває повторні виявлення. Кожен етап цього процесу – це складний конвеєр, він повинен бути точно налаштований самостійно і отримана система дуже повільна, займаючи більше 40 секунд на зображення під час тестування (рис. 7.15).

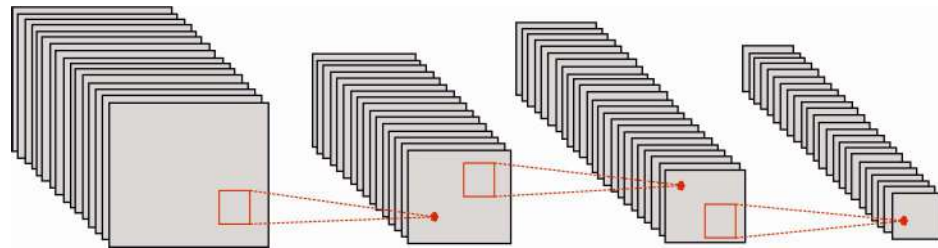


Рисунок 7.15 – Приклад розбиття зображення на рівні (R-CNN)

YOLO має певну схожість з R-CNN. Кожна клітинка пропонує потенційні обмежувальні рамки та оцінює їх бокси з використанням згорткових ознак. Проте наша система накладає просторові обмеження на пропозиції осередку сітки, які допомагають пом'якшити багаторазове виявлення одного і того ж об'єкта. Система також пропонує набагато менше обмежувальних квадратів, лише 98 на зображення в порівнянні з приблизно 2000 із вибіркового пошуку. І останнє, дана система поєднує ці окремі компоненти в єдину, спільно оптимізовану модель;

– MultiGrasp [378]. Принцип YOLO за дизайном схожий на роботу виявлення захоплення за Redmon. Сітковий підхід YOLO подібний до прогнозу обмежувальної рамки заснований на системі MultiGrasp для регресії до захоплень. Однак виявлення захоплення – це багато простіше завдання, ніж виявлення об'єктів. Потрібно лише щоб MultiGrasp передбачив єдину область для охоплення зображення, що містить один об'єкт. Для цього не потрібно оцінювати розмір, розташування, або межі об'єкта або передбачити його клас, потрібно знайти лише регіон, придатний для захоплення. YOLO передбачає обидва обмеження бокса та ймовірності класів для кількох об'єктів, кількох класів на зображенні (рис. 7.16).

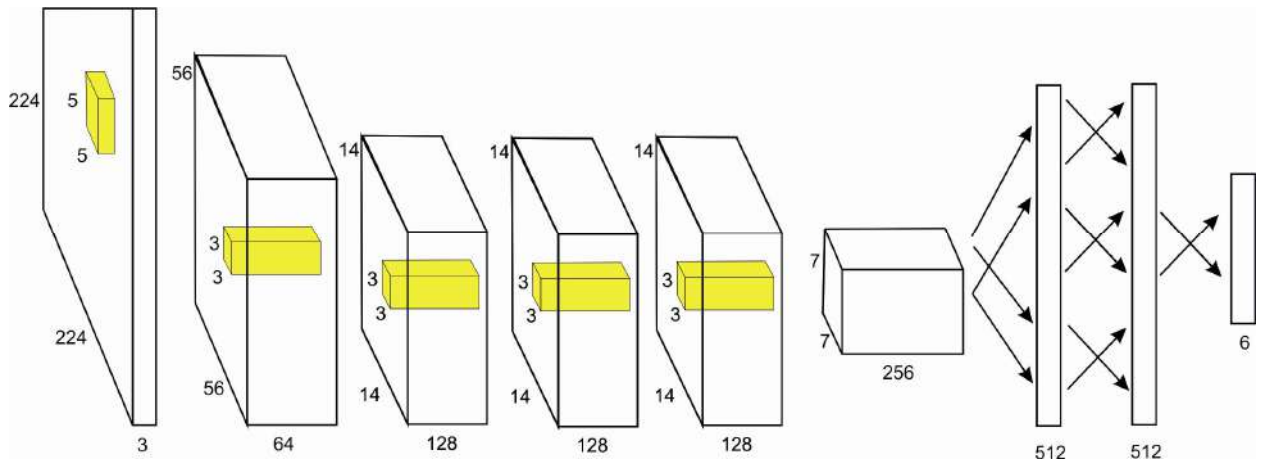


Рисунок 7.16 – Приклад розбиття зображення на рівні (MultiGrasp) [29]

7.4 Проведення експерименту ідентифікації об'єктів методом YOLO

Даний етап містить в собі розуміння яким чином працює система YOLO, написання програми ідентифікації предметів які знаходяться на зображенні.

Одразу можемо зауважити, що на початковому етапі вже враховується те, що середовище для розробки – є налаштоване, а саме:

- налаштування середовища для написання програми (PyCharm, Brackets або Notepad++);
- створено документ з розширенням .names;
- побудовано файл конфігурації YOLO, розширення .cfg;
- побудовано системний файл YOLO, розширення .weights;
- камера NexiGo N960E 1080P 60FPS підготовлена до роботи;
- завантаження бібліотеки OpenCV в систему.

Функція, яка присутня в розробці програми – це так званий прийом «краплі» (або «blob»):

```
blob = cv2.dnn.blobFromImage(frame, 0.00392, (320, 320), (0, 0, 0), True,
crop=False)
```

```
net.setInput(blob)
```

```
outs = net.forward(output_layers)
```

Дана функція дозволяє полегшити попередню обробку зображень для класифікації глибокого навчання [8].

Наступний крок – зробимо тест нашого зображення, а саме поділимо його на декілька рівнів, збільшимо крок вибірки. Рисунок 7.17, демонструє тільки поділ нашого зображення без урахування системи YOLO.



Рисунок 7.17 – Поділ зображення на декілька рівнів

Етапом для досягнення цілі служить підключення файлів конфігурації YOLO та пошук всіх схожих елементів, які існують в нашій базі. Відбувається процес зіставлення елементів, тобто асоціація. На рисунку 7.18

продемонстровано код програми та отриманий результат, з урахуванням пошуку елементів.

Важливо зауважити, в даній роботі використовується третя версія YOLO(3). З аналізу рисунка 7.18 видно, що наша програма тільки показала де ці об'єкти знаходяться, але не показала який вони мають розмір. Система працює швидко і одразу показує, що на зображенні є ініціалізація трьох елементів з чотирьох (філіжанка, ноутбук та мишка до нього) [5].

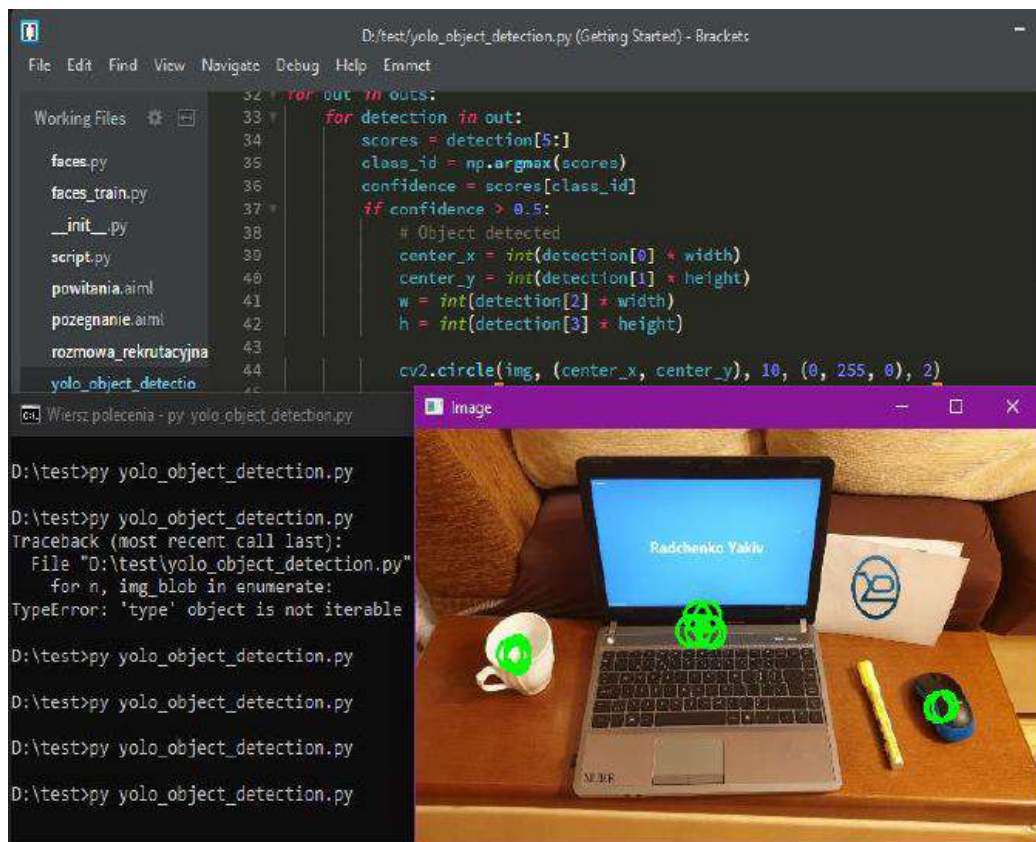


Рисунок 7.18 – Підключення файлів конфігурації YOLO, знаходження об'єктів на зображенні

Пропонуємо вдосконалити нашу програму і додати межі цих об'єктів (рис. 7.19). Внизу вихідного зображення продемонстровано рядок коду зв'язку OpenCV з побудовою цих меж на кожному елементі, їх розмірність та вибраний колір, який ініціалізує ці об'єкти.

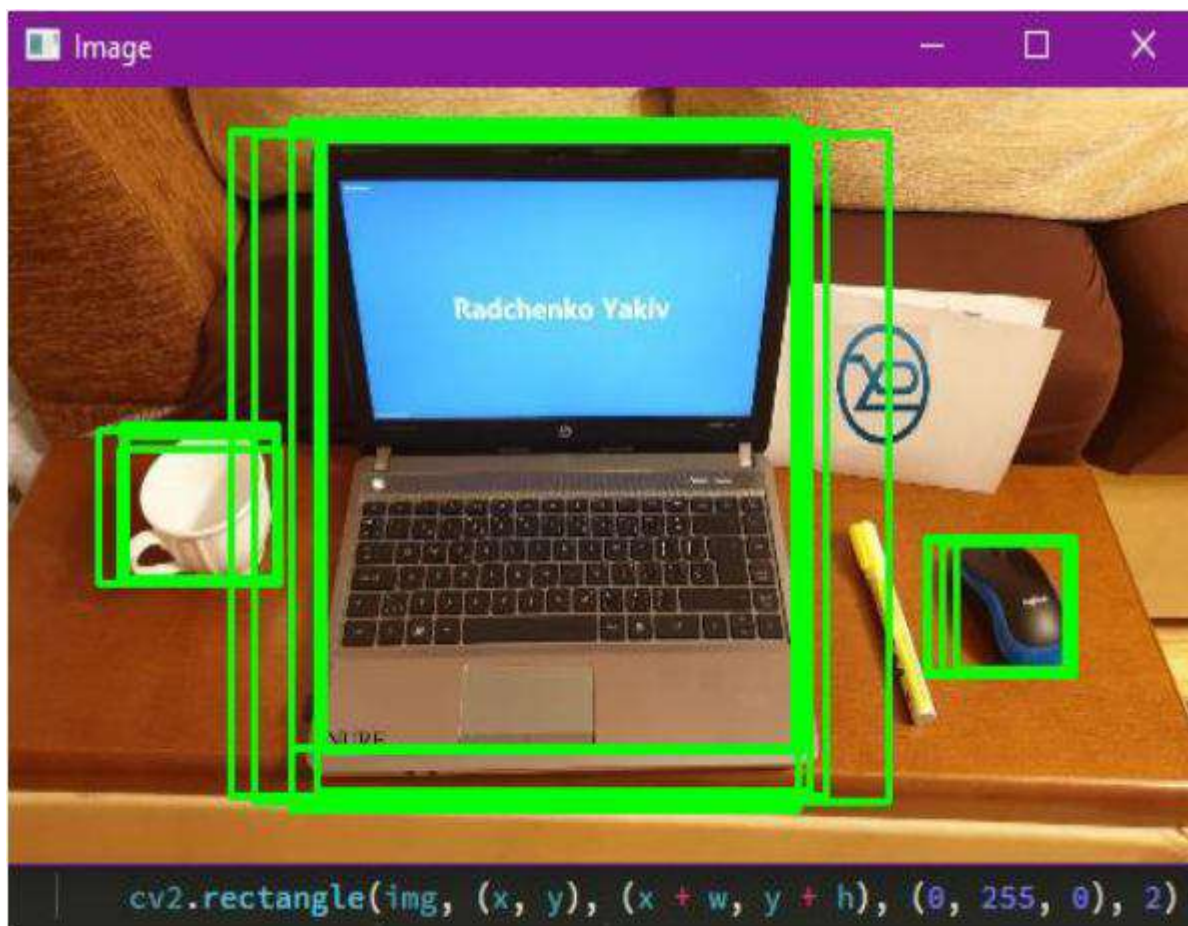


Рисунок 7.19 – Конфігурація параметрів для знаходження всього контуру об'єкта

В який спосіб з'являються обрамлення на об'єктах можемо побачити в коді програми:

```
x = int(center_x - w / 2)
y = int(center_y - h / 2)
boxes.append([x, y, w, h])
confidences.append(float(confidence))
class_ids.append(class_id)
```

Де, x та y беруть на себе позиціонування об'єкта відносно довжини та широти. Наступним чином показано, що елемент буде в середині так званого «бокса» і має свій власний ID, який дає ініціалізацію об'єкту.

Остатнім кроком в даній роботі – є правильна ідентифікація об’єктів і саме на цьому етапі використовується правила асоціацій та пошук всіх зв’язаних елементів на зображенні (рис. 7.20).

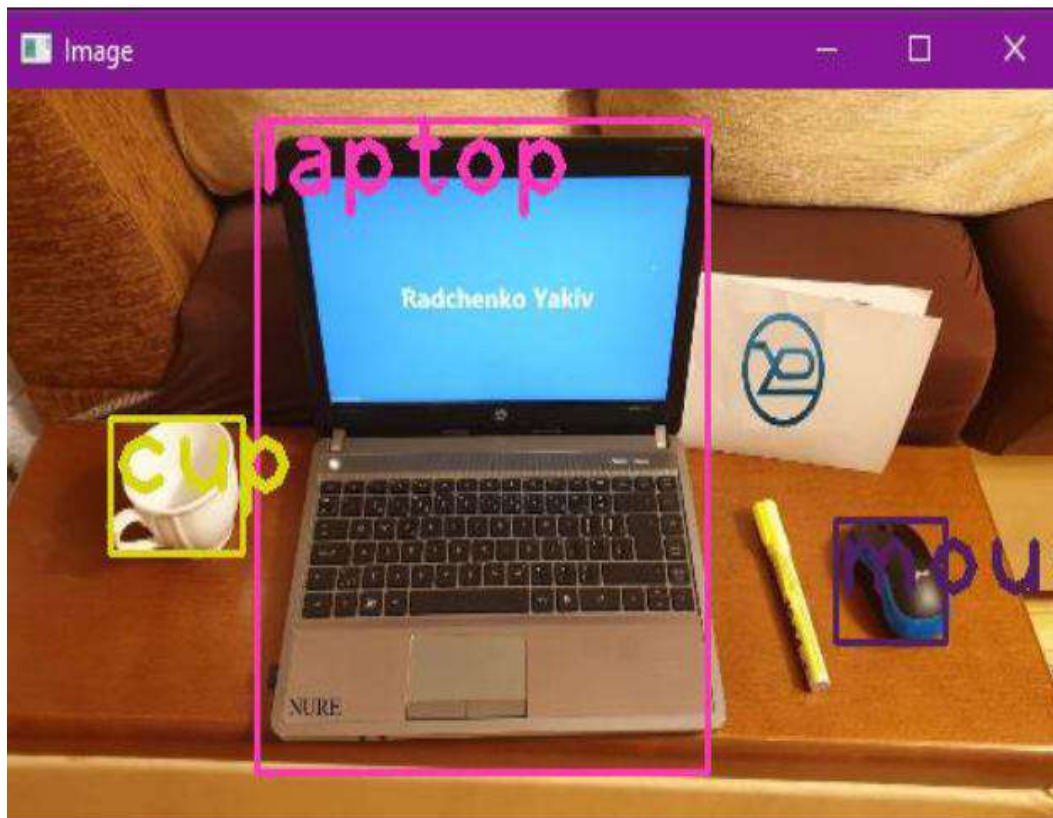


Рисунок 7.20 – Ідентифікація об’єктів на зображенні з використанням YOLO

Прив’язка до вже ініціалізованих об’єктів відбувається автоматично, приклад коду є нижче:

```
label = str(classes[class_ids[i]])
confidence = confidences[i]
color = colors[class_ids[i]]
cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)
cv2.putText(frame, label + " " + str(round(confidence, 2)), (x, y + 30), font,
3, color, 3)
```

З отриманого результату можна зробити висновок, а саме: система YOLO накладає сильні просторові ліміти на обмеження передбачення блоків,

оскільки кожна клітинка сітки передбачає лише два блоки і може мати лише один клас. Це просторова локалізація обмежує кількість найближчих об'єктів, яких наша модель може передбачити. Іншими словами, YOLO бореться з дрібними об'єктами, які об'єднуються у групи, наприклад зграя птахів.

Важливий недолік даної системи – поки ми тренуємося на функції втрат, яка наближається до ефективності виявлення, наша функція втрат обробляє помилки в малих обмежувальних прямокутниках порівняно з великими боксами. Невелика помилка у великому боксі як правило, є доброякісною, але як невелика помилка у малому боксі має набагато більший вплив на IOU. Наше головне джерело помилок – неправильна локалізація.

Даний етап містить в собі експериментальну частину вдосконалення нашої програми, в плані роботи в реальному часі.

На рисунку 7.21 продемонстровано набір різних кухонних приборів, камера робить зйомку зверху (приблизно $< 90^{\circ}$).



Рисунок 7.21 – Вивід отриманого зображення з камери (вид зверху $< 90^{\circ}$)

Рисунок 7.22 демонструє результат роботи програми та показує похибку в пошуку об'єктів та правильність ідентифікації. Програма розуміє, що на зображенні знаходяться такі об'єкти:

- ніж;
- виделка;
- філіжанка;
- ложка десертна / ложка звичайна.



Рисунок 7.22 – Отриманий результат програми

Даний дослід виявить залежність між часом потрібним для знаходження потрібних об'єктів та їх поправну ідентифікацію [14].

На рисунку 7.23 продемонстровано такий самий набір кухонних приборів, камера робить зйомку збоку (приблизно $< 45^\circ$). Це без урахування нашої програми.



Рисунок 7.23 – Вивід отриманого зображення з камери (вид збоку $< 45^\circ$)

Рисунок 7.24 демонструє результат роботи програми і показує вже інший результат в пошуку об'єктів та правильності ідентифікації.



Рисунок 7.24 – Знаходження всіх об'єктів та поправність ідентифікації

Можемо побачити, що під кутом у $< 45^\circ$ програма працює як треба і робить пошук всіх елементів, які тільки можуть знаходитися на отриманому зображенні.

Останній етап дослідження покаже залежність від кута зйомки. На рисунку 7.25 продемонстровано такий самий набір кухонних приборів, камера робить зйомку збоку (приблизно $<15^{\circ}$).



Рисунок 7.25 – Вивід отриманого зображення з камери (вид збоку $<15^{\circ}$)

Рисунок 7.26 демонструє результат роботи програми та показує похибку зйомки від кута, пошук об'єктів та правильність ідентифікації.

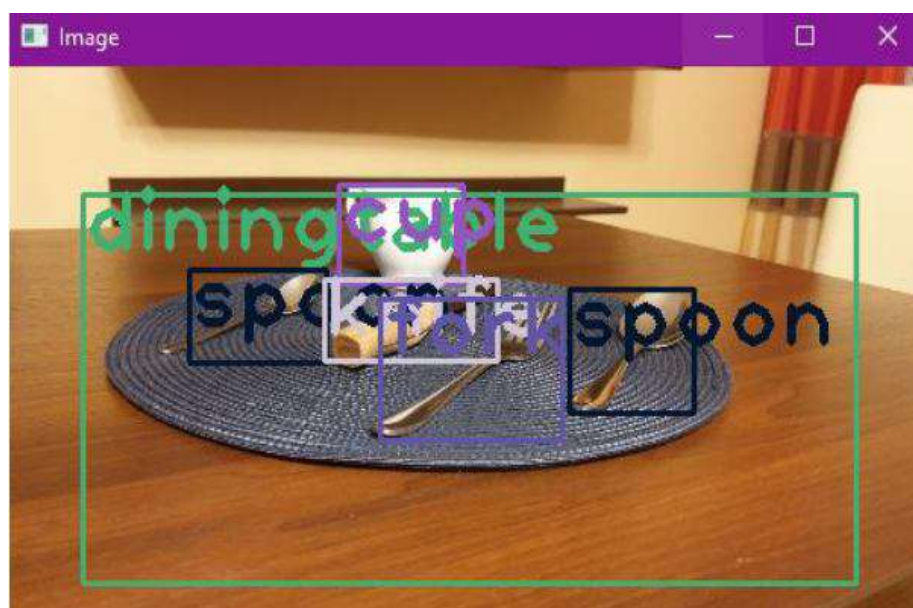


Рисунок 7.27 – Вивід отриманого зображення з камери (вид збоку $<15^{\circ}$)

На наступному рисунку 7.28 буде продемонстровано звичайне знаходження об'єктів та подальша їх ідентифікація.

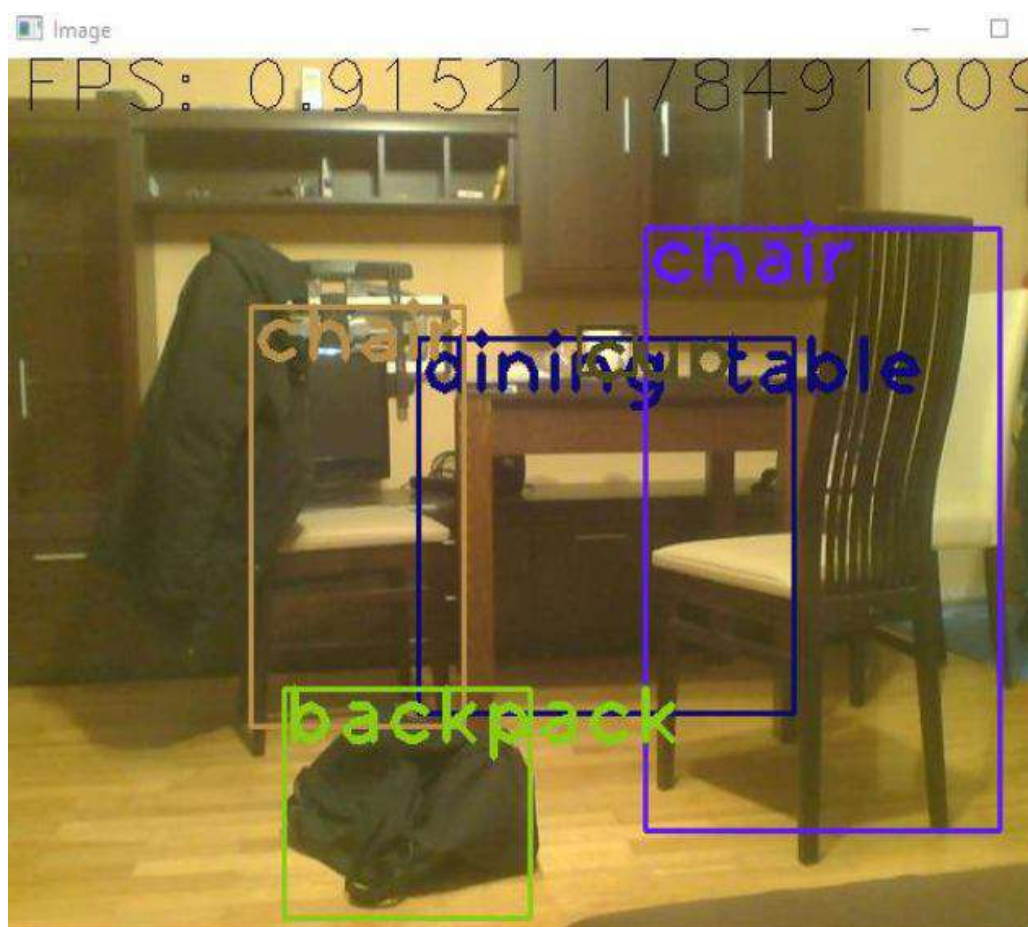


Рисунок 7.28 – Знаходження об'єктів в кімнаті та поправна їх ідентифікація

8. ГРУПОВЕ УПРАВЛІННЯ МОБІЛЬНИМИ РОБОТАМИ

8.1 Проблематика групового управління у мобільній робототехніці

Одиночний МР успішно використовується в різних галузях виробництва, проте спектр виконуваних завдань такого робота невеликий, так як існують такі технічні обмеження [379, 380]:

- обмежені енергоресурси і, як наслідок, малий радіус дії;
- обмежена функціональність (кількість виконуваних функцій МР) через необхідність використання енергоефективного програмно-апаратного базису, а також мінімізації набору корисного навантаження;
- високі ризики при функціонуванні в екстремальних умовах (в агресивних середовищах), так як вихід з ладу одиночного МР унеможливорює подальше виконання завдання.

Найбільш ефективним підходом до вирішення даних проблем є використання разом кількох роботів для виконання поставленого завдання, тобто застосування РТС на базі групи роботів. В рамках подібної системи роботи за допомогою групової взаємодії здатні ефективно вирішувати ряд складних завдань, нездійсненних для одиночного робота, наприклад: комплексне дослідження територій (завдання SLAM) [381], складання складних конструкцій, транспортування великогабаритних та багатоскладових вантажів, виконання бойових завдань, тощо.

Можливість виконання подібних завдань за допомогою РТС на базі групи роботів обумовлена низкою ключових переваг перед поодиноким МР:

- більший радіус дії, що досягається за рахунок розосередження роботів по всій робочій зоні;
- розширений набір виконуваних функцій у разі використання гетерогенної групи роботів, тобто роботів із різним функціональним призначенням;

– вища ймовірність виконання завдання в екстремальних ситуаціях (наприклад, у космосі, під водою або в зоні радіоактивного забруднення).

Ця перевага обумовлена можливістю перерозподілу завдань між роботами у групі у разі виходу з ладу деяких з них.

Тим не менш, при створенні РТС, що має подібні переваги, також виникає ряд проблем управління, які пов'язані з організацією групової взаємодії роботів.

У випадку вирішення завдання групового управління можна уявити як знаходження і реалізацію системою управління (СУ) такої послідовності узгоджених дій роботів у групі, що забезпечує досягнення поставленої мети. При цьому важливо враховувати, в якому середовищі дана РТС функціонує. Якщо параметри середовища відомі (детерміноване середовище), можна заздалегідь визначити послідовність узгоджених дій (технологічних операцій) для досягнення мети, наприклад, при автоматизації складального процесу декількома роботами-маніпуляторами.

Якщо РТС функціонує в умовах недостатності інформації про навколишнє середовище (недетерміноване середовище), то СУ визначає послідовність дій учасників групи у досягненні поставленої мети. Ці умови функціонування особливо актуальні для РТС на основі МР, при виконанні таких завдань, як картографія, розмінування, розвідувальні та рятувальні роботи в зонах техногенних аварій тощо [382].

У цьому випадку при вирішенні завдання групового управління, крім визначення послідовності дії для роботів групи, виникають такі проблеми:

- забезпечення комунікації між агентами розподіленої групи;
- оптимальний розподіл завдань з урахуванням функціонального призначення агентів (якщо об'єкт управління – гетерогенна група МР);
- забезпечення безпечного пересування агентів у загальному робочому просторі.

Таким чином, можна зробити висновок, що однією з ключових проблем, що виникають при функціонуванні групи МР у невизначеному

середовищі, є планування безпечних траєкторій руху, що дозволяють будувати безаварійні маршрути, близькі до оптимальних (найкоротших), що підтверджується і аналізом дослідницьких праць [383- 385].

8.2 Завдання пошуку оптимальної траєкторії

Існує безліч алгоритмів, що дозволяють вирішувати задачу пошуку траєкторії відповідно до заданих критеріїв. Проте, більшість цих алгоритмів є модифікацією «базових» методів планування шляху, оптимізованих під певні умови.

Умовно «базові» алгоритми пошуку шляху можна поділити на 3 групи [386-388]:

- 1) алгоритми простого обходу перешкод;
- 2) методи пошуку шляху за графом;
- 3) інтелектуальні алгоритми.

Застосування того чи іншого типу алгоритму обумовлюється відповідними критеріями, які визначають ефективність застосування даного алгоритму на вирішення поставленого завдання управління.

У рамках розв'язання задачі планування траєкторій руху для групи МР можна виділити такі критерії, що визначають ефективність застосування будь-якого алгоритму планування:

- висока якість побудови траєкторії, тобто алгоритм дозволяє формувати траєкторії, близькі до оптимальних (найкоротші чи близькі до найкоротших, в умовах даної конфігурації робочої зони);
- помірне витрачання апаратних ресурсів МР;
- простота формалізації завдання планування (простота математичного уявлення зміни робочої зони);
- швидкодія (швидкість розрахунку траєкторії) в умовах складної конфігурації робочого простору (за наявності великої кількості перешкод чи перешкод складної форми);

– можливість ефективного застосування для планування траєкторії в умовах зміни конфігурації робочої зони, що динамічно змінюється (за наявності рухливих перешкод, а також інших МР).

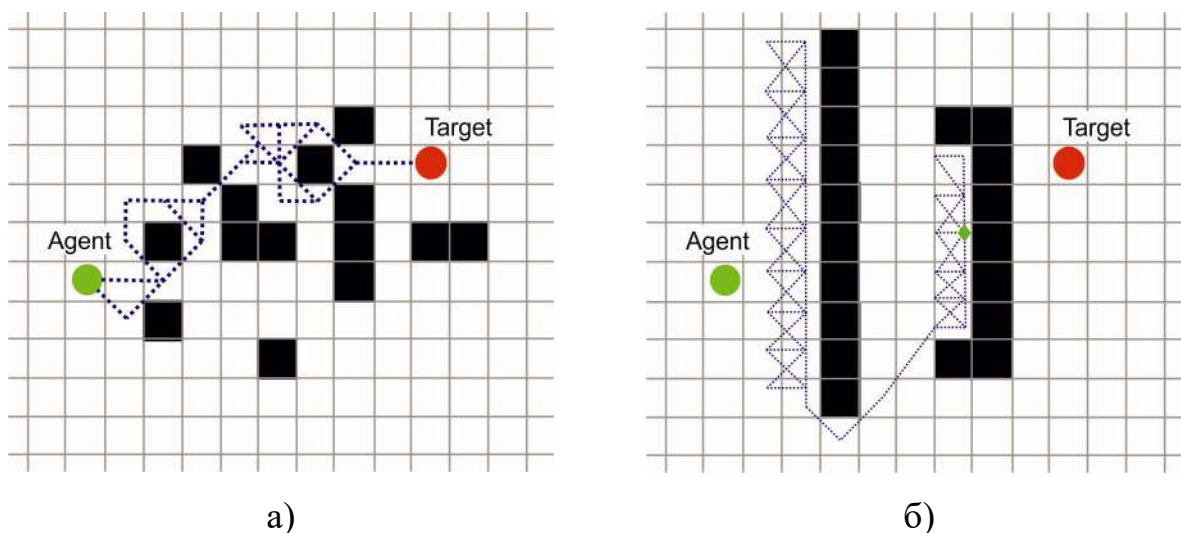
Проведемо аналіз типових алгоритмів пошуку шляху відповідно до заданих критеріїв.

Алгоритми обходу перешкод. Основною проблемою під час пошуку шляху є обхід перешкод. Найбільш простим способом вирішення даної проблеми є ініціювання маневру обходу при безпосередньому зіткненні з перешкодою або при наближенні до нього на мінімально можливу відстань, що визначається датчиками.

Для застосування даної методики обходу перешкод в алгоритмі необхідно відстежувати два основних параметри: умова виявлення перешкоди (наприклад, певне показання бортового далекоміра) та відносне положення агента та його цілі. Виходячи з цих умов, можлива реалізація наступних стратегій обходу [389,390]:

– зміна напрямку руху у разі виявлення перешкоди. При цьому в пріоритеті знаходяться напрямки, які скорочують відстань між агентом і метою (за умовною прямою лінією). Таким чином, агент «прагне» до мети, обминаючи перешкоди за допомогою невеликих переміщень (пропорційних габаритам агента) за будь-якими можливими напрямками. Даний алгоритм може давати прийнятні результати під час роботи з простими перешкодами (рис. 8.1, а). Тим не менш, при роботі з перешкодами складнішої форми може знадобитися занадто багато часу для знаходження шляху обходу, або шлях може бути взагалі не знайдений у разі зациклювання (рисунок 8.1, б).

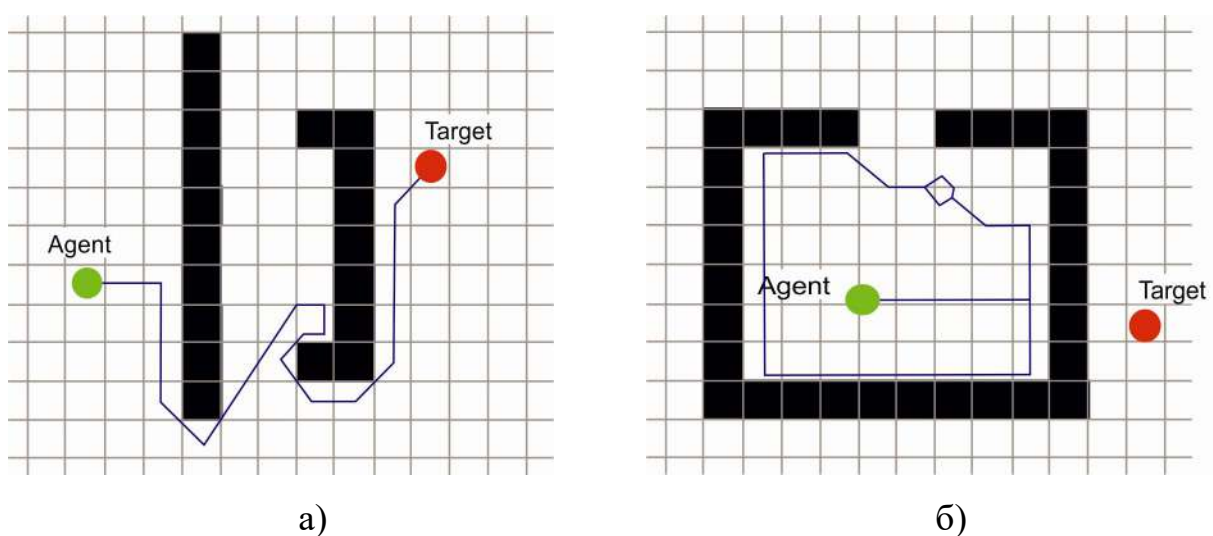
Трасування перешкод. Цю стратегію можна використовувати для обходу перешкод складної форми. Суть полягає в тому, що агент при виявленні перешкоди починає наслідувати його контур (трасувати) на певній відстані, використовуючи, наприклад, малопотужні бортові далекоміри.



а) з простими перешкодами; б) випадок зациклювання

Рисунок 8.1 – Переміщення у випадковому напрямку [27]

Трасування також здійснюється з урахуванням відстані (по прямій) до мети. Цей алгоритм успішніше справляється з перешкодами складної форми, ніж при стратегії випадкової зміни напрямків (рис. 8.2, а), проте, ймовірність зациклювання у певних випадках також висока – наприклад, при сценарії типу «пошук виходу з кімнати» (рис. 8.2, б).



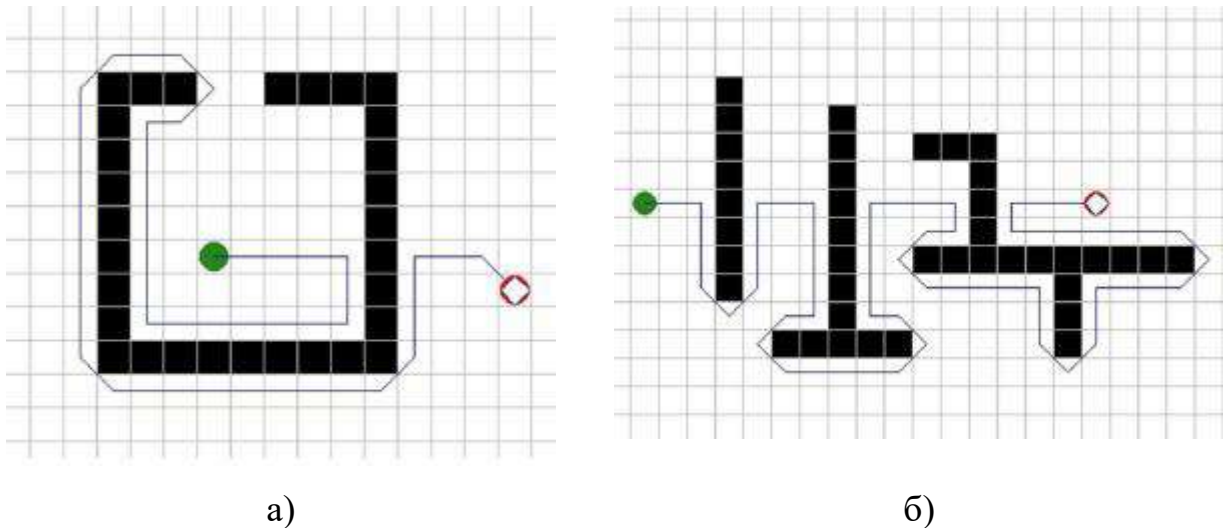
а) для стратегії випадкової зміни напрямків;

б) для сценарію типу "пошук виходу з кімнати"

Рисунок 8.2 – Трасування перешкод

«Надійне трасування» перешкод. Ця методика є модифікацією алгоритму трасування [391]. Суть роботи модифікованого алгоритму полягає в тому, що зі стартової позиції агента обчислюється пряма лінія до цілі, агент рухається по цій лінії до виявлення перешкоди, при цьому трасування виявленої перешкоди проводиться до того моменту, поки дана лінія не буде перетнута знову.

Дана модифікація гарантує знаходження обхідного шляху до цілі (рис. 8.3, а). Основним недоліком даної стратегії є те, що агент може продовжувати трасування до перетину контрольної лінії, навіть коли шлях до цілі вільний (рис. 8.3, б).



а) стратегія знаходження обхідного шляху до цілі; б) сценарії коли агент продовжує трасування, коли шлях до цілі вільний

Рисунок 8.3 – Надійне трасування перешкод

Тим не менш, алгоритм може бути використаний у комбінації зі звичайним трасуванням і використовуватися у разі зациклювання. Незважаючи на те, що розглянуті вище алгоритми обходу перешкод можуть успішно вирішувати завдання пошуку шляху до цілі, можливі ситуації, в яких необхідне планування траєкторії руху агента. Наприклад, таке планування необхідно при пошуку шляху близького до оптимального (за часом або довжиною) в умовах складної конфігурації робочого простору,

тобто за наявності «зважених» (важкопрохідних) областей, рухомих (динамічних) перешкод чи інших агентів.

Базові алгоритми пошуку за графом. В теорії графів є кілька базових алгоритмів, які можна використовувати для вирішення завдання планування траєкторії за наявності складних перешкод і зважених областей. Дані алгоритми здійснюють пошук рішення шляхом проходів вузлами графа, які відповідають зміні стану. Таким чином, для формалізації завдання пошуку шляху використовується дискретизація робочого простору, а перехід з однієї дискретної клітини в іншу еквівалентний переходу між двома вузлами графа, тобто – зміні стану.

Існує велика кількість алгоритмів, які використовують різні стратегії переходу на вузли графа в процесі пошуку рішення.

Пошук у глибину (DFS) [392]. У цьому алгоритмі пошук здійснюється шляхом послідовного переходу на найближчий не відвіданий вузол «вглиб» нумерованого графа. Якщо знайдена вершина з усіма відвіданими сусідніми вузлами, то алгоритм запускається від попереднього вузла і так далі, поки не буде знайдена ціль (або не залишиться не відкритих вузлів).

У класичному вигляді алгоритм, в основному, застосовується для пошуку за графами з деревоподібною структурою. При роботі з графом, який описує дискретний робочий простір (є «сітчною» структурою), необхідно передбачити окремий параметр для обмеження «глибини» пошуку (алгоритм DLS) [393]. Дана модифікація дозволить скоротити кількість ітерацій та виключити зациклення.

Незважаючи на те, що алгоритм дозволяє гарантовано знаходити шлях до цілі, але процес пошуку може вимагати багато часу у випадку, якщо обрано неправильний напрямок пошуку. Також алгоритм гарантує отримання оптимального рішення – найкоротшого шляху (рисунок 8.4):

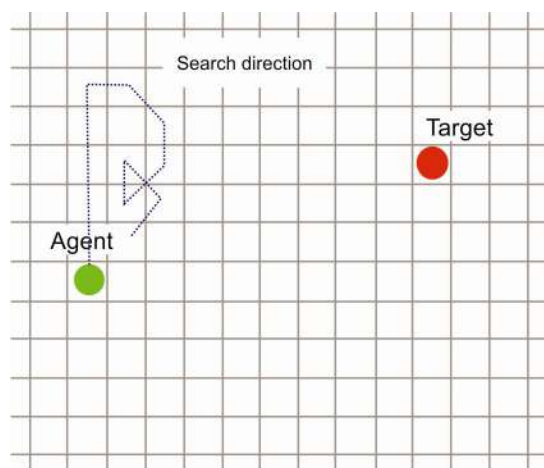


Рисунок 8.4 – Пошук в глибину

Для усунення зазначених вище недоліків можна скористатися модифікованою версією пошуку DLS – **алгоритмом з ітеративним заглибленням під час пошуку (IDDFS)**. Ця стратегія пошуку шляху по графу передбачає ітеративне збільшення глибини пошуку (починаючи з одиниці) і при кожній ітерації запускається алгоритм DLS. Основні етапи алгоритму IDDFS [394]:

1. Встановлюється глибина пошуку, що дорівнює одиниці: $d = 1$.
2. Від поточного (або стартового) вузла запускається пошук у глибину DLS із встановленою глибиною d .
3. Якщо ціль не знайдена, то збільшуємо глибину пошуку на одиницю ($d = d + 1$) і переходимо до п. 2.

Данный алгоритм решает проблему определения правильной глубины поиска, а также позволяет находить наиболее оптимальное решение среди всех переборных алгоритмов по времени и памяти.

Даний алгоритм вирішує проблему визначення правильної глибини пошуку, а також дозволяє знаходити найбільш оптимальне рішення серед усіх алгоритмів перебору за часом і пам'яті. Тим не менш, через збільшення глибини пошуку на кожній ітерації, застосування алгоритму даного типу для пошуку траєкторії руху в умовах робочого простору, що динамічно змінюється, дуже важке.

На другому етапі здійснюється відновлення шляху за допомогою зворотного пошуку (починаючи від цільового вузла) вузлів з найменшим значенням маркера (рис. 8.6):

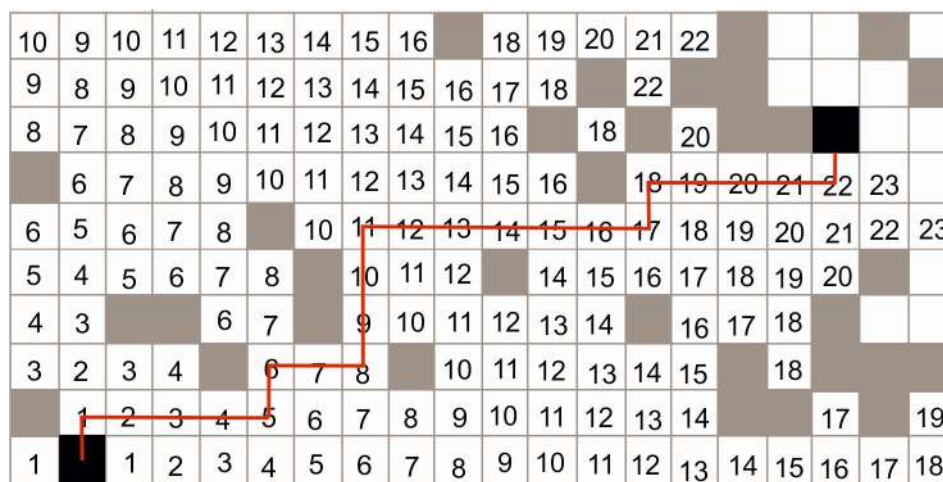


Рисунок 8.6 – Процес відновлення колії

Цей алгоритм здатний гарантовано знаходити найкоротший шлях до цілі, його часто використовують для трасування друкованих плат.

Проте, хвильовий алгоритм має і недоліки, основний з них: неможливість коректного врахування діагональних переходів. При поширенні хвилі маркер призначається лише найближчим вузлам за прямими переходами, якщо ж маркувати діагональні вузли як найближчі, то при відновленні шляху довжина діагонального переходу все одно не буде врахована і кінцеве рішення може бути неоптимальним. Також алгоритм здійснює пошук за вузлами у всіх напрямках, тобто витрачає зайвий час на переходи у зворотному напрямі від цільового вузла.

Частково вирішити цю проблему може використання модифікованого алгоритму: двонаправлений пошук в ширину. Модифікація полягає в тому, що відновлення шляху відбувається одночасно зі стартового та кінцевого вузлів до перетину. Використання даної модифікації може прискорити процес пошуку (до 2 разів), проте, у великих робочих просторах у плані

ресурсомісткості алгоритм не є найефективнішим, також залишається проблема з оптимальністю знайденого шляху.

Алгоритм Дейкстри. Даний алгоритм був розроблений відомим нідерландським ученим Едсгером Дейкбуд ще наприкінці 50-х років минулого століття, але досі цей метод пошуку (і його модифікації) залишається одним із найпопулярніших при плануванні шляху за графом [396].

Відмінною особливістю алгоритму є те, що при уявленні дискретного робочого простору кожному ребру графа привласнюється значення, еквівалентне довжині ("вартості") переходу між вузлами, з'єднаними даним ребром. Таким чином, для плаского робочого простору з ортогональною дискретизацією кожен вузол має 8 найближчих вузлів, з якими він з'єднаний ребрами за 8 можливими напрямками: 4 – для прямих переходів з «вартістю» 1, наприклад, 10, і ще 4 – для діагональних переходів з «вартістю» 14 відповідно.

Для кожного нового вузла та його сусідів обчислюється «вартість» переходу, яка є найменшою сумою всіх переходів, починаючи від стартового. На кожному кроці алгоритм шукає необроблені вузли та обчислює відстань до всіх його сусідів, при цьому, якщо шлях до одного із сусідніх вузлів вже був розрахований на ранніх етапах, це значення може бути оновлено, якщо на поточному етапі отримана «вартість» шляху буде меншою. Дані операції продовжуються доти, доки не буде знайдено мінімальну довжину шляху до цільового вузла. На рисунку 8.7 показано приклад роботи алгоритму Дейкстри у дискретному робочому просторі.

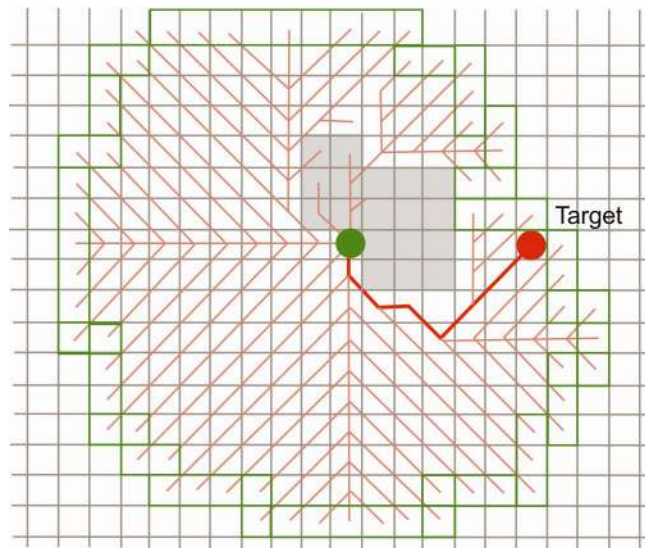


Рисунок 8.7 – Алгоритм Дейкстри

Основною перевагою даного алгоритму і те, що він враховує можливі діагональні переходи. Також, виходячи з того, що довжини ребер можна призначати залежно від конфігурації робочого простору, можна вирішувати проблему зважених областей і будувати траєкторії, близькі до найкоротших.

Тут також варто згадати алгоритм Беллмана-Форда, який (аналогічно алгоритму Дейкстри) здійснює пошук найкоротшого шляху зваженим графом [397]. Алгоритм дещо повільніший в окремих випадках за алгоритм Дейкстри, але дозволяє працювати з графами, що мають ребра з негативними значеннями ваг. Також цей алгоритм активно застосовується для динамічної мережевої маршрутизації (наприклад, протокол RIP) [398].

Тим не менш, для коректної роботи алгоритму Дейкстри потрібна велика кількість пам'яті для зберігання значень відкритих вузлів, а також необхідно багаторазово виконувати процедури пошуку та порівняння цих значень зі значеннями, наведеними на наступних ітераціях для отримання найменшого. Всі ці обчислювальні операції уповільнюють процес розв'язання задачі.

Також цей алгоритм має той самий недолік, що й хвильовий алгоритм: процес пошуку йде у всіх напрямках, без урахування розташування цілі, що негативно позначається на часі розв'язання задачі. Однак цей ефект також

може бути корисним для вирішення певних завдань, наприклад, при пошуку перетину зон покриття радіосигналу. Проблема «спрямованості» пошуку можна вирішити з допомогою алгоритму A^* .

Алгоритм A^* . Даний алгоритм вважається одним із кращих для вирішення задачі пошуку шляхів, близьким до оптимальних [399]. Формалізація дискретного робочого простору і принцип побудови шляху в A^* відбувається аналогічно алгоритму Дейкстри: складається граф, вузли (n) якого з'єднуються ребрами з відповідними довжині переходу «вагами». Насамперед проглядаються вузли, суміжні зі стартовим, і обирається той їх, який має мінімальне значення «вартості» (довжини) переходу $f(n)$, тощо. Головна особливість полягає в тому, що при розрахунку $f(n)$ враховується параметр «близькості» вузла n до цільового вузла:

$$f(n) = g(n) + h(n) \quad (8.1)$$

де $g(n)$ – найменша сумарна вартість шляху у вузол n з точки старту;

$h(n)$ – евристичне наближення вартості шляху до мети від вузла n .

Евристичне наближення може бути обчислено у різний спосіб, але найшвидшим з них вважається функція «манхеттенської відстані», де величина $h(n)$ дорівнює сумі модулів різниць координат поточного та цільового вузлів.

З огляду на евристичне наближення, алгоритм переглядає насамперед ті вузли, які «здаються» провідними до цілі. На рисунках 8.8, а та 8.8, б показано вплив величини $h(n)$ на напрямок перегляду вузлів: на рисунку 8.8, а при мінімальному значенні $h(n)$ напрямок пошуку схожий на алгоритм Дейкстри; на рисунку 8.8(б) зі збільшенням евристики $h(n)$ пошук став «сфокусованим». Таким чином, ефективність пошуку буде сильно залежати від того, наскільки величина $h(n)$ близька до реального значення «вартості»

При програмній реалізації алгоритму необхідно передбачити пріоритетний список (список Open) для зберігання множини вузлів, що становлять маршрути з початкової точки до всіх ще нерозкритих вершин графа. Вибір пріоритетного вузла кожного наступного переходу визначається за значенням $f(n)$. Алгоритм продовжує свою роботу до тих пір, поки значення $f(n)$ для цільового вузла не буде найменшим у порівнянні зі значеннями інших вузлів у черзі, інакше пошук зупиниться, коли всі вузли графа будуть відкриті. Усі відкриті вузли зберігаються в окремому списку *Closed*, щоб унеможливити їх повторну обробку під час пошуку.

Алгоритм A^* дозволяє ефективно вирішити задачу пошуку найкоротшого шляху до цілі, але при цьому для застосування даного алгоритму існує ряд обмежень:

- для обробки карт великої розмірності (кілька тисяч дискретних осередків і більше) обсяг даних у списках Open і Closed також зростатиме, що в умовах обмеженості бортових апаратних ресурсів МР може негативно позначитися на швидкості пошуку та значно збільшити час на розв'язання задачі розрахунку траєкторії;

- оскільки алгоритм для кожного нового вузла обчислює мінімальну сумарну «вартість» шляху (починаючи від стартового), то можлива некоректна робота алгоритму при роботі в умовах робочого простору, що динамічно змінюється, або групового планування траєкторій, де необхідно враховувати додаткові правила вирішення конфліктних ситуацій між агентами;

- методика обчислення евристичного наближення $h(n)$ дуже впливає на якість роботи алгоритму.

Незважаючи на зазначені обмеження A^* , даний алгоритм є одним з найбільш ефективних класичних алгоритмів планування траєкторії та активно застосовується для управління рухом МР.

Крім розглянутих вище методик пошуку шляху, існують інші підходи, які використовують подібні графам структури, у яких вузли можуть також

враховувати інші стани МР (наприклад, кут орієнтації, швидкість і прискорення робота). Проте, подібні методики найактуальніші для планування за умов статичного середовища, так як побудова подібних графів або дерев станів вимагає повноти інформації про робочий простір, яку проблематично надати заздалегідь, наприклад, за наявності інших МР у тій же області.

8.3 Методика планування траєкторії за допомогою нейронної карти

8.3.1 Поняття нейронної картки. Основна ідея методу планування за допомогою нейронних мереж (НМ) Хопфілда полягає в тому, щоб використовувати нейронну карту як динамічне уявлення двовимірного дискретного робочого простору [400]. Таким чином, використовувана для вирішення задачі планування НМ матиме певну топологію, а кількість нейронів в мережі буде дорівнює кількості дискретних осередків робочого простору.

Нехай є деяке зовнішнє джерело інформації (локальна сенсорна система МР, віддалений сервер і т.д.), на підставі якої можна, в межах заданої дискретизації, визначити місцезнаходження МР або розташування цілі та перешкод. Тоді, використовуючи цю інформацію, можна сформувати вектор вхідних сигналів даної НМ. Якщо якесь із вхідних значень відмінно від нуля, то в мережі запускається процес активації: через прямі та зворотні зв'язки сигнал поширюється по всій НМ до тих пір, поки не буде досягнуто стан стійкості (по Ляпунову).

Якщо вихідний вектор мережі в стані стійкості подати у вигляді матриці, розмірність якої збігається з топологічним уявленням дискретного робочого простору, тобто кількість елементів матриці буде збігатися з кількістю дискретних осередків, то отримана матриця і буде «нейронною картою», яку надалі можна використовувати при розрахунку траєкторії до заданої цілі.

8.3.2 Архітектура системи планування траєкторії на базі НМ Хопфілда

Для опису принципу роботи системи планування траєкторії з урахуванням методу «нейронних карт» розглянемо МР, що знаходиться у двовимірному дискретному робочому просторі (С). Система планування траєкторії для одиночного МР складається з двох основних блоків: нейронна карта та конструктор шляху (КШ). Архітектура системи показана рисунку 8.10.

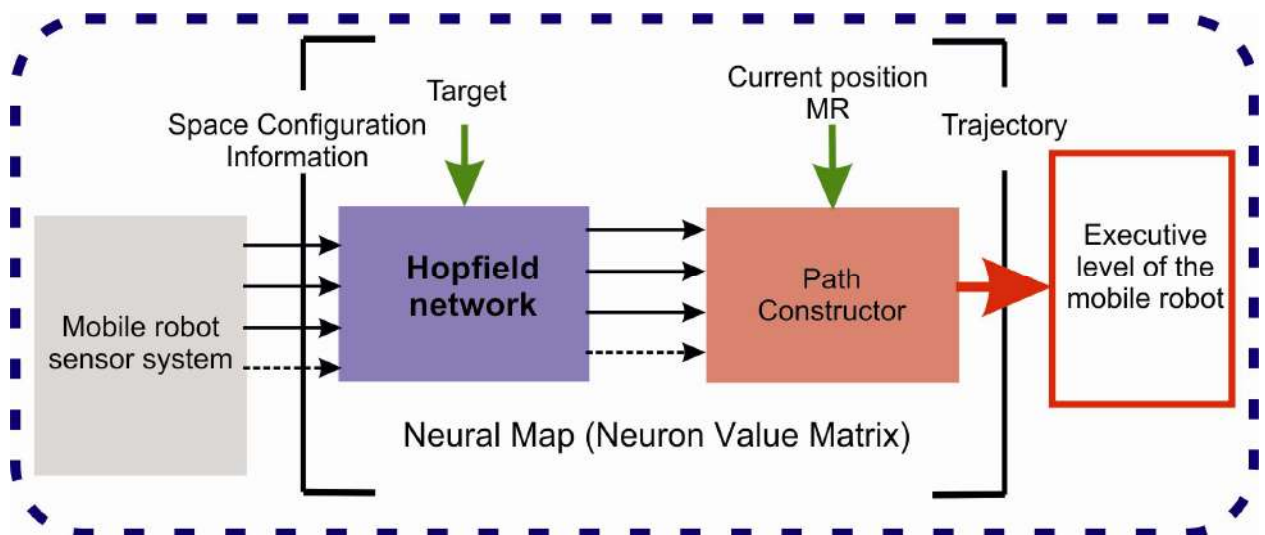


Рисунок 8.10 – Архітектура системи планування траєкторії на основі нейронної карти

Система працює за наступним алгоритмом:

- визначаються ціль та поточна позиція МР;
- на підставі сенсорної інформації про С визначається конфігурація робочого простору (розташування перешкод);
- формується вхідний сигнал для НМ Хопфілда, при цьому вхідний сигнал нейрона, що відповідає цільовому осередку, буде більшим 0 (точка активації);
- очікується завершення процесу активації;
- формуються матриці вихідних сигналів (нейронна карта);

- нейронна картка подається на блок КШ;
- отримання кінцевої траєкторії руху до заданої цілі.

8.3.3 Системні характеристики

Розглянута система планування траєкторії (рис. 8.10) відноситься до класу технічних систем. За призначенням її можна класифікувати як керуючу, оскільки вона забезпечує збирання, переробку та передачу інформації, необхідної для прийняття глобальних рішень та виконання локальних завдань (зокрема, завдання позиціонування). Як і будь-якій системі, їй притаманні такі базові системні властивості:

1) *Подільність* – систему можна уявити такою, що складається з ряду підсистем або елементів: НМ Хопфілда, що формує карту і КШ і будує по ній траєкторію; у свою чергу, НМ складається з безлічі взаємопов'язаних базових елементів нейронів і т.д.

2) *Цілісність* – властивості всієї системи залежить від властивостей її елементів, отже зміна властивостей елементів тягне за собою зміну властивостей всієї системи. Наприклад, при зміні архітектури НМ, нейронна карта буде різною при однаковій конфігурації робочого простору, що спричинить необхідність зміни функцій розрахунку траєкторії блоку КШ.

3) *Емерджентність* – властивості системи-цілого не є сумою властивостей її елементів, таким чином, розв'язання задачі планування траєкторії неможливе без системоутворюючих інформаційних зв'язків між елементами, що забезпечують перетворення та структурування вхідних та вихідних даних для обробки відповідними елементами (зокрема, зв'язок між джерелом інформації) і НМ, зв'язок НМ з КШ та зв'язок конструктора з підсистемою управління робота).

4) *Інтегративність* – визначається наявністю системоутворюючих та системозберігаючих факторів, найважливішими серед яких є неоднорідність та несуперечність її елементів. В аналізованій системі дані чинники багато чому

визначаються збіжністю НМ, яка, у свою чергу, залежить від коректності вхідних даних, які надходять від джерела інформації.

5) *Комунікативність* – система не ізольована, вона взаємодіє із середовищем; середовище та система мають вплив один на одного. Взаємодія системи з середовищем відбувається як на початковому етапі розв'язання задачі планування траєкторії (при отриманні інформації про конфігурацію робочого простору через джерело інформації), так і в процесі формування кінцевого рішення (при русі робота по розрахунковим крокам траєкторії).

6) *Ієрархічність* – система, з одного боку, складається з підсистем, а з іншого боку, сама є частиною більш загальної системи (наприклад, частиною СУ групою МР).

Виходячи з того, що використовується в системі НМ Хопфілда має властивість запам'ятовування образів («пам'ять»), сформовані нейронні карти можуть бути використані багаторазово і виступати як деяка база знань, на підставі якої система будуватиме або коригуватиме траєкторію в процесі руху агента («приймати рішення»). На основі цієї властивості можна також охарактеризувати систему як *інтелектуальну*.

Також аналізована система, безсумнівно, належить до класу складних систем. На приналежність до цього класу вказує складність функціонування (поведінки) даної системи, яка багато в чому визначається безліччю можливих станів НМ Хопфілда та нейродинамікою.

Нейродинамічні процеси, у свою чергу, обумовлюються як особливостями структури базових обчислювальних елементів мережі (нейронів), так і структурної організації всієї мережі (топологією).

8.4 Математична модель НМ Хопфілда

Для аналізованої у складі системи планування НМ Хопфілда нейрон i характеризується вхідним вектором γ ; сигналом власного зворотного зв'язку v_{ii} ; вихідними сигналами пов'язаних нейронів v_{jin} , де n – розмірність мережі;

матрицею вагових коефіцієнтів W_i ; сумарним (мережевим) сигналом v_i ; функцією активації Φ ; вхідним сигналом I_i ; власним вихідним сигналом стану v_{out} . Приклад структури нейрона НМ Хапфільда представлений на рисунку 8.11

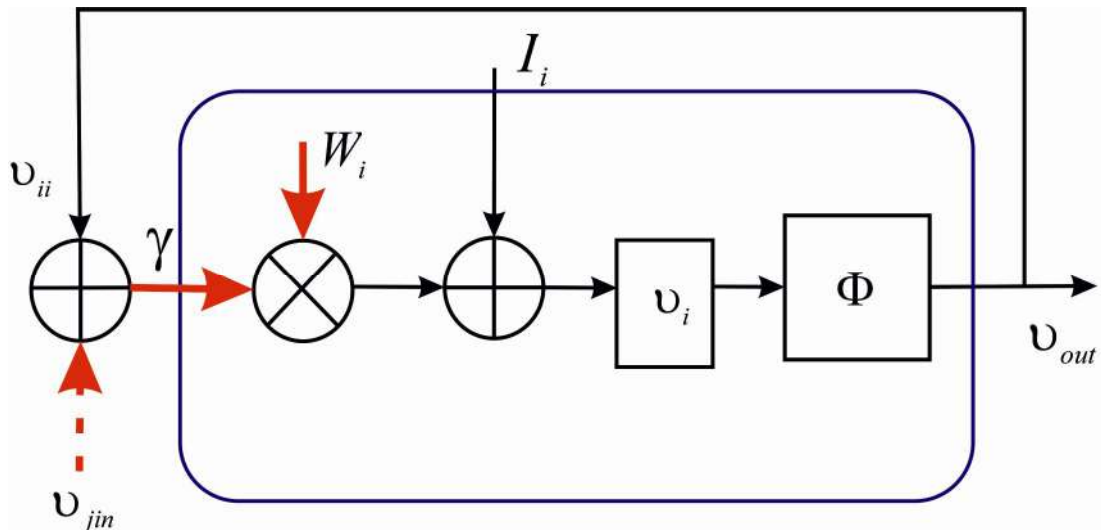


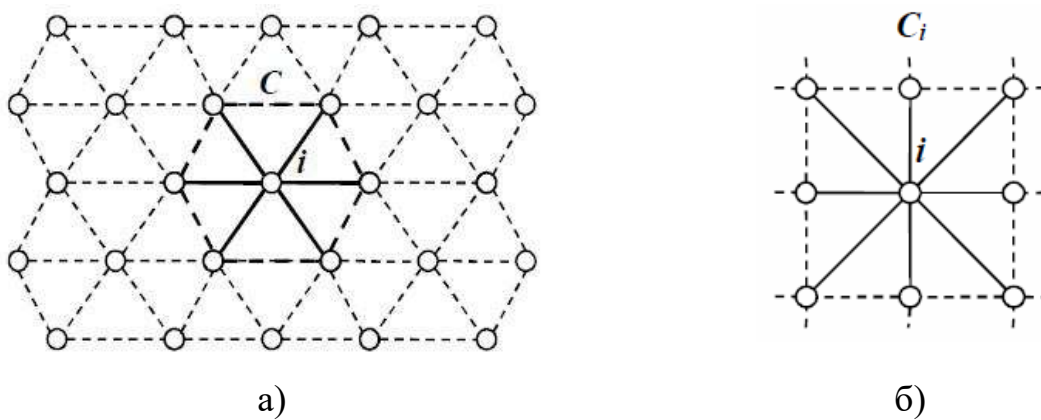
Рисунок 8.11 – Структура нейрона НМ Хапфільда

Власний вихідний сигнал нейрона v_{ii} та вихідні сигнали v_{jin} пов'язаних (зворотними зв'язками) з i нейронів утворюють вхідний вектор сигналів γ , який, у свою чергу, зважується матрицею вагових коефіцієнтів W_i , складається вхідним сигналом (зміщенням) I_i і формує скаляр v_i , який проходить через нелінійну функцію активації Φ :

$$v_i = W_i \times \gamma(t) + I_i(t) \quad (8.1)$$

Функція активації $\Phi(v)$ визначає граничні значення сигналів нейронів i , отже, характер енергетичної взаємодії між усіма нейронами. Таким чином, дана функція багато в чому визначає безліч можливих станів мережі i , відповідно, збіжність.

Якщо розглянути плоский робочий простір C , рівномірно розділений на n дискретних осередків, то відповідна НМ Хопфілда буде також складатися з n нейронів, рівномірно розподілених (образно) по центрах дискретних осередків даного простору. Кожен нейрон матиме зв'язки (як прямі, і зворотні) лише з нейронами з суміжних осередків. Таким чином, мережа буде упорядкованою структурою у вигляді «решітки». Залежно від заданої геометричної форми дискретних осередків НМ матиме різну топологію. Приклад топологій наведено на рисунку 8.12.



а) гексагональна; б) ортогональна

Рисунок 8.12 – Приклад топології мережі

Як видно з рисунка 8.12, кожен нейрон взаємодіє тільки з сусідніми в межах своєї підмножини («домена») C_i . Ця особливість забезпечує «хвильовий» характер поширення сигналу в мережі.

У межах даної роботи приймемо ортогональну дискретизацію робочого простору як основну. Таким чином, модель НМ Хопфілда, що розглядається, також матиме ортогональну топологію, де кожен «домен» складатиметься з дев'яти нейронів: центральний і 8 сусідніх, відповідно (за 8-ми можливими напрямками руху).

НМ Хопфілда відноситься до класу мереж, що самоорганізуються, для яких застосовується методика навчання «без вчителя». Ця методика заснована на так званому правилі Хебба [401]: синаптичний зв'язок, що

з'єднує два нейрони, буде посилюватися, якщо в процесі навчання обидва нейрони узгоджено відчують збудження або гальмування. Виходячи з даного правила, вихідний сигнал аналізованої СМ формуватиметься мережею самостійно, за умови ненульового вхідного сигналу, і визначатиметься конфігурацією зв'язків між нейронами та передатною функцією нейронів.

Подальшу обробку вихідних даних НМ та розрахунок траєкторії виконує окрема обчислювальна підсистема – «конструктор шляху». Залежно від наявного обсягу інформації про конфігурацію робочого простору блок КШ може виконувати розрахунок усієї траєкторії (якщо розташування перешкод відомо повністю і дана конфігурація робочого простору статична), або розрахунок окремих частин траєкторії (розташування перешкод невідомо заздалегідь, тобто конфігурація робочого простору динамічна).

Обчислення траєкторії КШ починається після формування матриці стану НМ (нейронної карти) від стартової позиції агента. На кожному новому етапі обчислюється 8 значень – градієнти ($grad_{ij}$) за всіма можливими напрямками від поточного значення нейрона i до суміжних (j -х нейронів). Далі вибирається максимальне значення градієнта ($grad_{max}$) і здійснюється перехід у цьому напрямку на нову позицію ($i = j$). Пошук триває до тих пір, поки не буде досягнуто цільового нейрона з максимальним значенням енергії (E_{max}). Узагальнений алгоритм роботи КП представлений рисунком 8.13.

При побудові системи розрахунку траєкторій для групи МР у загальному робочому просторі на КШ також покладається ключова функція вирішення конфліктних ситуацій з урахуванням пріоритетів та відповідного динамічного коригування траєкторії.

Залежно від розв'язуваних завдань, а також інформаційних і обчислювальних ресурсів, можливі різні підходи до реалізації системи планування групи МР. Нижче (рис. 8.13) розглянемо апаратні та програмні особливості побудови даної системи за різних підходів.

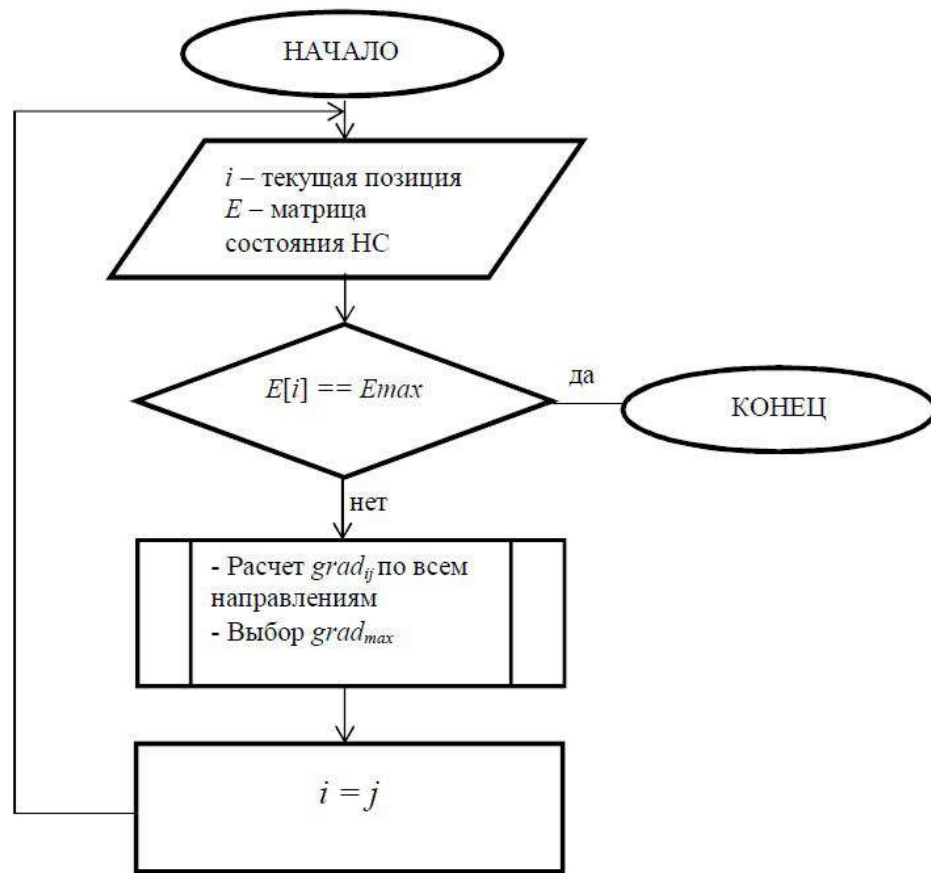


Рисунок 8. 13 – Узагальнений алгоритм роботи конструктора шляху

8.5 Розробка програмно-апаратної архітектури системи планування для групи мобільних роботів

Вибір підходу до реалізації системи планування визначається за такими основними критеріями:

- обчислювальні можливості апаратної бази;
- конфігурація робочої зони (розмір, розташування перешкод);
- вимоги до швидкості вирішення завдання позиціонування;
- вимоги до енергоємності (кількість витрачених мобільним агентом енергоресурсів на розрахунок та перехід на відстань, що дорівнює одиниці дискретного робочого простору);
- вимоги до надійності системи.

Виходячи з цих критеріїв, можливі такі варіанти побудови системи планування [402]:

- централізований;
- гібридний;
- децентралізований.

Розглянемо ці підходи більш детально.

Централізований підхід. Побудова системи планування за централізованим варіантом передбачає вирішення завдання позиціонування групи у єдиному центрі. Таким чином усі обчислювальні операції системи планування виконуються у центральному багатопоточному обчислювальному блоці (ЦП). ЦБ за наявною інформацією про конфігурації робочого простору виконує формування нейронних карт для кожного з агентів, а також розрахунок траєкторій з урахуванням швидкостей та вирішення можливих конфліктних ситуацій.

Варто також зазначити, що обчислювальні операції НМ та КШ для кожного агента виконуються паралельно в окремих потоках з доступом до загальних програмних ресурсів. Агенти, у свою чергу, лише приймають каналами даних керуючі сигнали і відповідно до них приводять у дію виконавчі механізми, тобто беззастережно виконують «вказівки» центру. Реалізація системи планування централізованого підходу представлена рисунку 8.14.

До основних переваг реалізації цієї системи планування можна віднести:

- простота реалізації апаратної частини;
- легкість збільшення продуктивності всієї системи за рахунок збільшення продуктивності апаратної частини ЦП;
- дешевизна;
- низька енергоємність агентів;
- висока швидкодія.

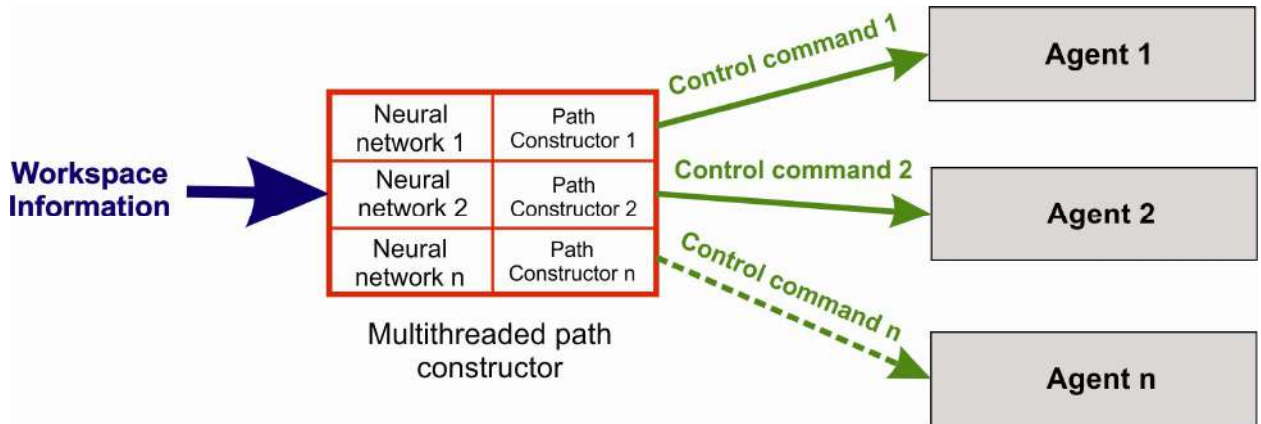


Рисунок 8.14 – Централізована система планування траєкторії на базі НМ Хопфілда

Відзначимо також і ряд недоліків:

- низька безвідмовність – вихід з ладу ЦП повністю зупиняє роботу системи (можливе часткове вирішення проблеми за допомогою резервування ЦП);
- високі вимоги до обчислювальних можливостей ЦП (при великій дискретизації робочого простору, тобто підвищення точності переміщень) і оперативної пам'яті (при великій кількості агентів);
- обмеженість розмірів загальної зони переміщень у зв'язку з необхідністю організації віддалених каналів зв'язку з ЦП і, як наслідок, з необхідністю встановлення потужних енерговитратних приймальних пристроїв на агентах;
- неможливість роботи у заздалегідь невідомій конфігурації робочої зони або за наявності динамічних перешкод;
- складність реалізації програмної частини, оскільки необхідно застосування технології многопоточного програмування, і навіть синхронізація обчислювальних потоків задля забезпечення доступу до спільних даних.

Гібридний підхід. При застосуванні гібридного підходу система планування реалізується із загальним блоком КШ та розподіленою НМ

Хопфілда. У цій конфігурації кожен агент оснащується бортовим обчислювальним блоком НМ і формує свою нейронну карту.

Сформовану нейронну карту агент по каналу зв'язку передає до загального центрального конструктора шляху (ЦКШ), який запускає процес розрахунку траєкторії в окремому потоці і здійснює видачу керуючих сигналів.

Керівні команди ЦКШ не підлягають до обов'язкового виконання агентом, так як у процесі руху агент може скоригувати або сформувати нову нейронну карту, в залежності від нової інформації, що надходить від сенсорів про робочий простір. Також, у разі виявлення перешкод на заданій траєкторії агент може самостійно обходити перешкоди з умовою повернення на задану траєкторію після здійснення маневру. Якщо ця умова не виконується і агент не здатний продовжувати рух по поточній траєкторії, то останні оновлені дані (поточні координати та карта) передаються в ЦКШ, який здійснює перерахунок траєкторії з урахуванням інформації від інших МР у складі групи.

Принцип реалізації системи планування за гібридною схемою представлений рисунку 8.15.

По відношенню до централізованого підходу цей принцип реалізації системи планування, крім подібних якостей, таких як: висока енергоефективність агентів, швидкодія та простота масштабування апаратної продуктивності системи загалом; також має такі ключові переваги:

- можливість роботи в заздалегідь невідомій конфігурації робочої зони або за наявності динамічних перешкод;
- значне зниження вимог до продуктивності апаратної частини системи зі збільшенням кількості агентів і збільшення дискретизації робочої зони, завдяки поділу нейрообчислень між агентами.

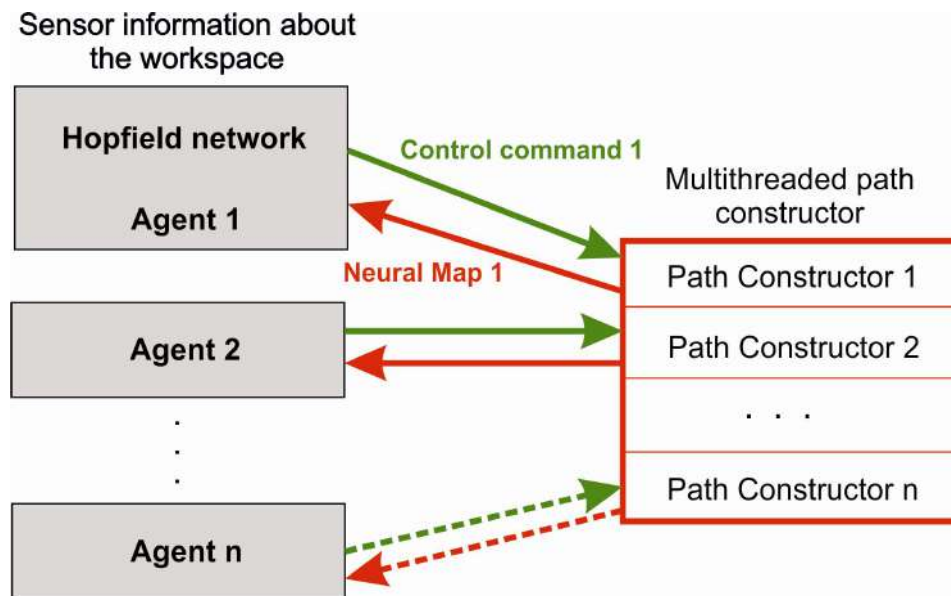


Рисунок 8.15 – «Гібридна» система планування траєкторії
на базі НМ Хопфілда

Тим не менш, зберігається і ряд недоліків, властивих системам з ознаками централізації:

- низька безвідмовність (можливе часткове вирішення проблеми за допомогою резервування ЦКШ);
- обмеженість розмірів загальної зони переміщень у зв'язку з необхідністю організації віддалених каналів зв'язку з ЦКШ і, як наслідок, з необхідністю встановлення потужних енерговитратних приймальних пристроїв на агентах;
- більш складне апаратне виконання учасників групи через наявність блоку обчислення НМ, а також необхідність встановлення бортових сенсорних пристроїв;
- складність реалізації програмної частини, так як необхідно застосування технології багатопоточного програмування та синхронізація обчислювальних потоків обробки даних з агентів у реальному масштабі часу.

Децентралізований підхід. При реалізації системи планування за децентралізованою схемою, кожен агент має власну систему планування траєкторії з обчислювальним блоком НМ і КШ на борту. Під час виконання завдання позиціонування кожен агент, за допомогою інформаційної взаємодії

бездротовими каналами зв'язку з іншими (найближчими) учасниками групи або координатором, самостійно коригує свою траєкторію і вирішує конфліктні ситуації. Реалізація децентралізованої системи планування траєкторій представлена рисунку 8.16.

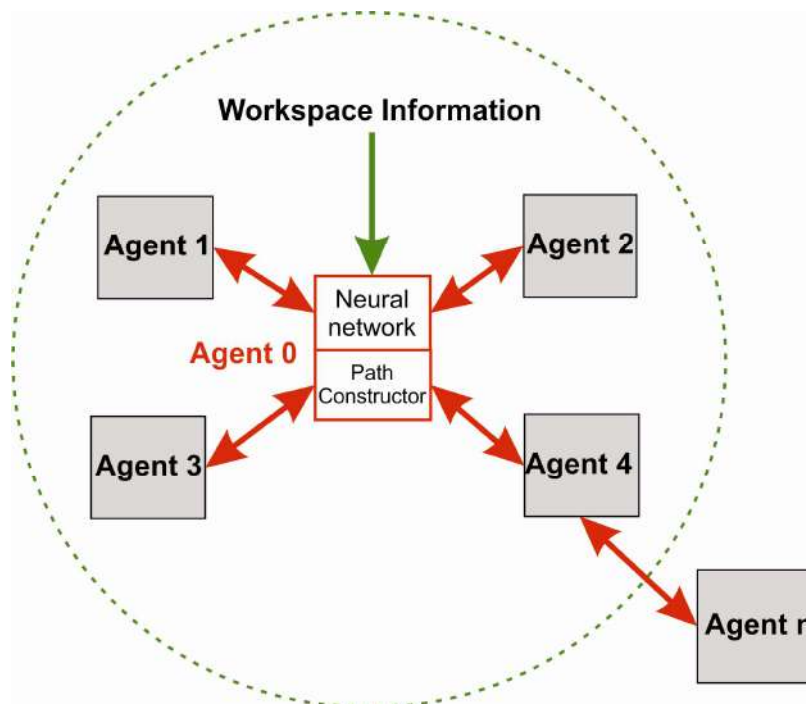


Рисунок 8.16 – Структурна схема розподіленої системи планування траєкторії

Розглянемо ключові переваги децентралізованого підходу під час реалізації системи планування траєкторій для групи МР:

- безвідмовність та надійність – вихід з ладу одного агента (або кількох) ніяк не впливає на вирішення завдання позиціонування для інших агентів;
- можливість вирішення завдань позиціонування в робочій зоні будь-якого розміру.

Незважаючи на очевидні переваги даного підходу, існує й низка недоліків:

– обмеженість обчислювальних ресурсів агента через особливо жорсткі вимоги до енергоефективності мобільних бортових комплексів і, як наслідок, нижчу швидкодію;

– складність і дорожнеча реалізації апаратної платформи агента.

Також варто зазначити, що система, реалізована за даним підходом, за складністю поведінки класифікуватиметься як самоорганізована, що визначається більш гнучкими реакціями системи на зовнішні впливи, а також розширеними можливостями адаптації та накопичення досвіду.

9 РОЗРОБКА ЕКСПЕРЕМЕНТАЛЬНОГО МАКЕТУ МОБІЛЬНОГО МАНІМУЛІЦІЙНОГО РОБОТА

9.1 Розробка системи керування мобільним маніпуляційним роботом

Система керування мобільною платформою спеціального призначення складається з великої кількості датчиків, виконавчих пристроїв і допоміжних модулів: підсистеми навігації, підсистеми управління живленням та інші. В якості предмета дослідження виступає мобільний робот з вдосконаленими маніпуляційними можливостями. Проведені дослідження [403-405] дозволили визначити, що основними вузлами робота є: вузол керування рухом, вузол керування маніпулятором, головний вузол керування, навігаційний вузол, сенсорний вузол, вузол забезпечення технічного зору, вузол керування живленням, вузол забезпечення зв'язку.

На рисунку 9.1 представлений експериментальний макет розробленого мобільного робота з маніпуляційними можливостями.

Враховуючи велику кількість різноманітних модулів, датчиків та виконавчих пристроїв використання централізованої системи керування є недоцільною.



Рисунок 9.1 – Розроблений мобільний робот

Для реалізації системи управління мобільним роботом пропонується використовувати операційну систему (ОС) на основі Robot Operating System (ROS), розроблену в 2007 році в Лабораторії Штучного Інтелекту Стенфордського Університету [406].

Особливістю ROS є підтримка стандартних служб операційних систем, таких як: апаратну абстракцію; низькорівневий контроль пристроїв; реалізацію функцій, що часто використовуються; передачу повідомлень між процесами, і управління пакетами.

ROS заснований на архітектурі графів, де обробка даних відбувається у вузлах, які можуть отримувати та передавати повідомлення між собою. Бібліотека орієнтована на Unix-подібні системи [407].

ROS має наступні сильні «сторони» для застосування його в даному дослідженні: сторона операційної системи ROS, як описано вище, та `ros-pkg` і набір пакетів, що підтримуються користувачами (організованих в набори, які називаються стек), реалізують різні функції робототехніки: SLAM, планування, сприйняття, моделювання та ін.

ROS випускається відповідно до умов BSD-ліцензії та з відкритим вихідним кодом. ROS безкоштовний для використання як з дослідницькою, так і комерційною метою. Пакети з `ros-pkg` поширюються за умов різних відкритих ліцензій.

Вибір ОС ROS дозволяє в подальшому проводити модернізацію системи управління мобільним роботом з можливістю нарощування функціональних можливостей та масштабування системи залежно від поставлених завдань.

На підставі даних припущень була розроблена система управління переміщенням мобільної платформи, структурна схема якої наведена на рисунку 9.2.

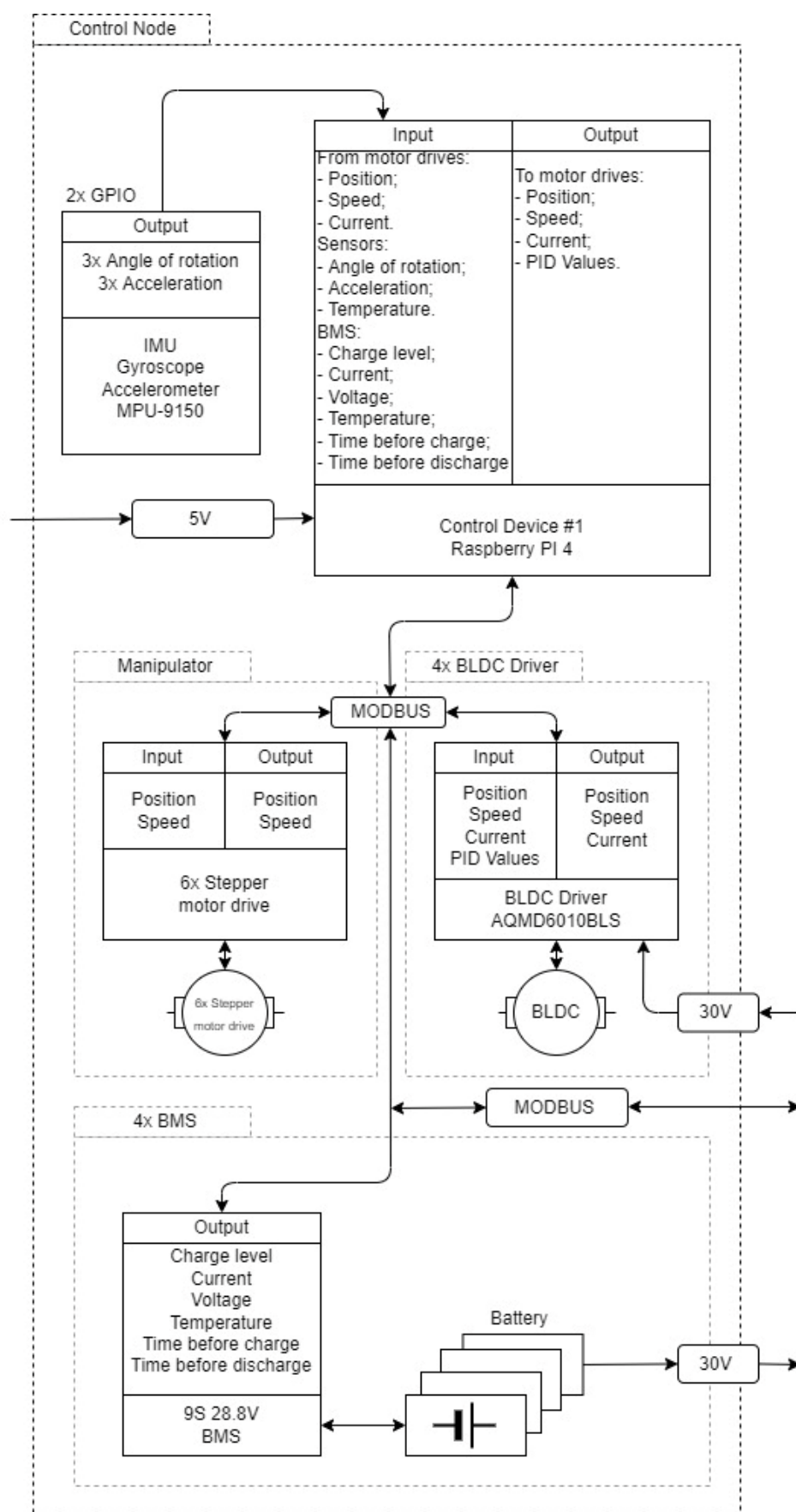


Рисунок 9.2 – Структурна схема управління переміщення мобільного робота

Структурна схема управління переміщенням мобільного робота (Control Node) складається з трьох основних блоків:

- Manipulator;
- Brushless DC electric motor (BLDS Driver);
- Battery Management System (BMS).

Розглянемо кожен блок детальніше. На рисунку 9.3 представлено структурну схему управління Manipulator.

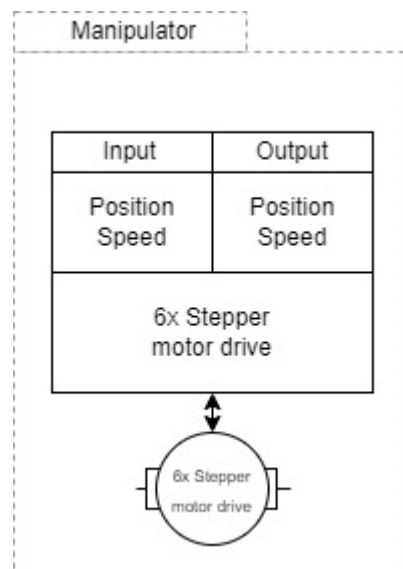


Рисунок 9.3. – Структурна схема керування маніпулятором

Як можна бачити з рис. 9.3 структурна схема управління складається з 6 крокових двигунів, які керуються за допомогою драйверів. Так само для оцінки параметрів переміщення, пропонується проводити контроль швидкості переміщення, що надалі дозволить домогтися високої точності переміщення, при цьому точно контролювати координати.

Структура BLDS Driver представлена на рис. 9.4. Структура реалізована на базі BLDC Driver AQMD6010BLS [408]. Загальний вигляд BLDC Driver AQMD6010BLS представлений на рисунку 9.5.

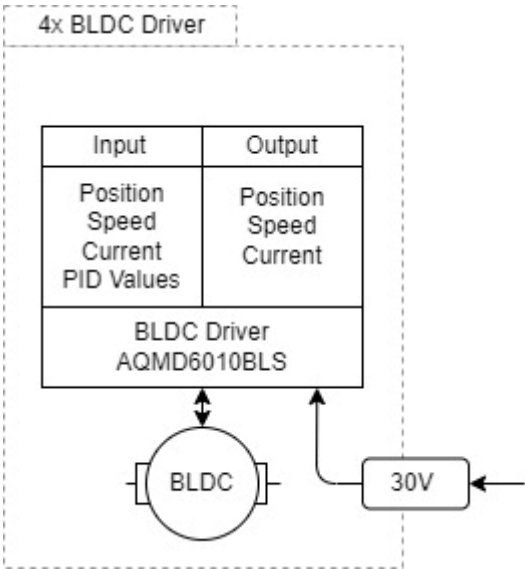


Рисунок 9.4 – Структурна схема BLDC блоку



Рисунок 9.5 – Загальний вигляд BLDC Driver AQMD6010BLS [408]

Основні характеристики BLDC Driver AQMD6010BLS представлені у таблиці 9.1.

Таблиця 9.1 – Основні характеристики BLDC Driver AQMD6010BLS

Параметри	Значення/Опис
1	2
Voltage range:	9-60V
Rated current:	10A
Maximum current:	12A

Control signals:	potentiometer, analog, PWM, pulse, frequency, switch, RS485
Speed regulation mode:	duty cycle, closed loop speed regulation, position control, torque control
Applicable motor:	DC sensorless brushless motor
Maximum matching motor:	60V-420W, 48V-335W, 36V-250W

Обраний драйвер дозволить контролювати такі параметри на вході:

- position;
- speed;
- current;
- PID Values

и на виході:

- position;
- speed;
- current.

Останній блок представленої структури (рис 9.6) – BMS. Заснований на базі складання високострумівих акумуляторів типу 18650 з використанням пристрою балансування при зарядженні [409]. Передбачається проводити контроль наступних вихідних параметрів:

- charge level;
- current;
- voltage;
- temperature;
- time before charge;
- time before discharge.

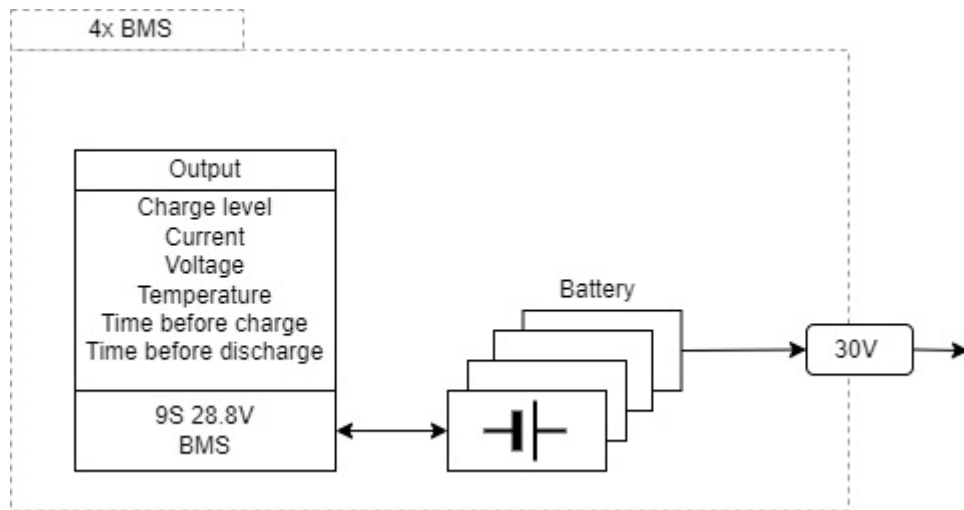


Рисунок 9.6 – Структура блоку BMS

Система управління мобільним роботом (Control Node) побудована з урахуванням одноплатного комп'ютера Raspberry Pi 3 Model B+ (див. розділ 3). Використовуючи датчик MPU-9150 [410], який підключається до Raspberry Pi 3 Model B+ по шині GPIO по інтерфейсу I²C. Дозволяє визначити наступні параметри мобільного робота:

- angel of rotation;
- acceleration.

Вхідними даними для Raspberry Pi є:

- від модуля управління двигунами: поточна позиція на основі даних від таходатчиків, струм, що споживається двигунами, швидкість руху;
- від сенсорів: кут обороту, дані від акселерометра, температура двигунів;
- від підсистеми контролю рівня заряду акумуляторів: рівень зарядки, струм, напруга живлення, температура, час, що минув з останньої зарядки і розрядки.

Для виконання завдання керування швидкістю руху використовується модуль `diff_drive_controller` із бібліотеки ROS. Цей модуль дозволяє керувати диференціальними приводами коліс робота. Управління виконується у вигляді команд підтримки заданої швидкості, що розділяються між двома колесами диференціальної пари. Від детекторів таходатчиків надходять

поточні дані одометрії для коригування поточної швидкості, залежно від алгоритму управління рухом.

Передача керуючих сигналів між усіма блоками мобільного робота будуть здійснюватися на базі протоколу MODBUS. Обґрунтуванням такого рішення стало те, що Modbus — відкритий комунікаційний протокол, заснований на архітектурі провідний — client-server. Широко застосовується у промисловості організації зв'язку між електронними пристроями. Може використовуватися для передачі даних через послідовні лінії зв'язку RS-485, RS-422, RS-232 та мережі TCP/IP (Modbus TCP). Також є нестандартні реалізації, що використовують UDP [411].

Для забезпечення заданої швидкості переміщення мобільного маніпуляційного робота передбачається використовувати PID – регулятор [412]. Структурна схема PID – регулятора швидкості для DC – двигунів представлена рисунку 9.7.

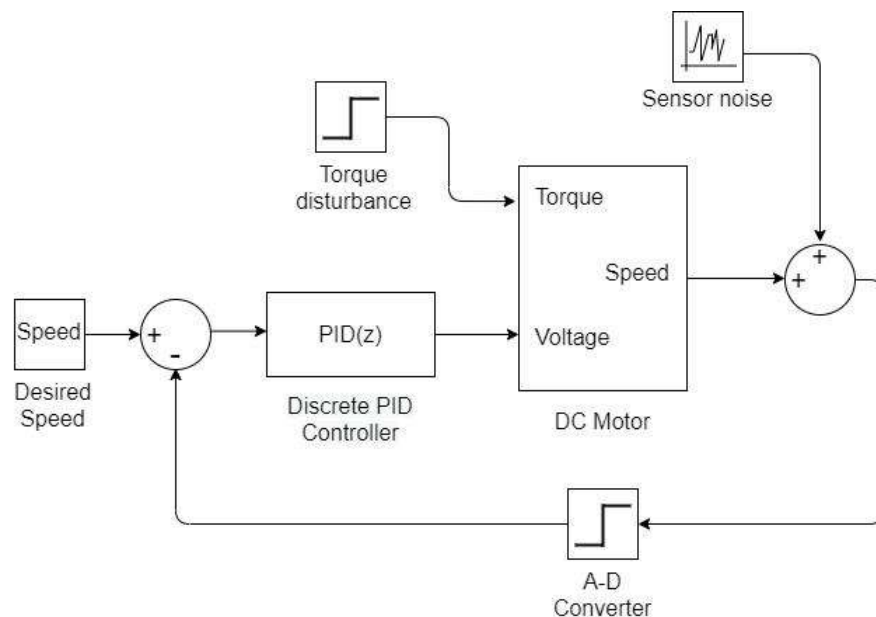


Рисунок 9.7 – Структурна схема PID-регулятора швидкості

При регулюванні швидкості використовуються дані про поточну швидкість, надані за допомогою сенсорів (Sensor Noise). Враховуючи похибку між виміряною швидкістю та бажаною, розраховується напруга на вході двигуна. У зворотну ланку включені блок моделювання шуму з

датчиками та A-D converter. Структурна схема Sensor Noise представлена рисунку 9.8.

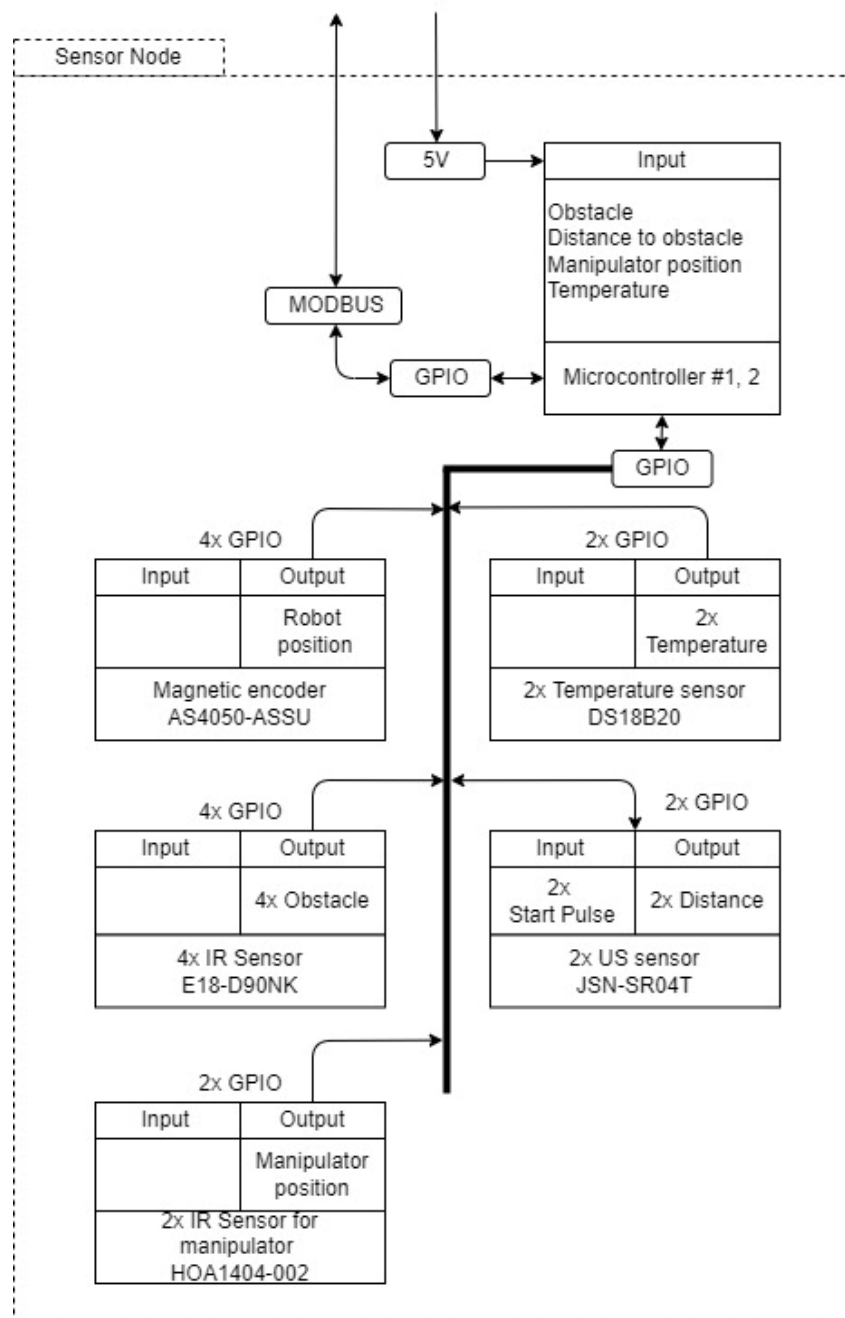


Рисунок 9.8 – Структурна схема Sensor Noise

Опис основних сенсерів і параметрів, які вони контролюють у МР, що розробляється, представлені в таблиці 9.2.

Таблиця 9.2 – Опис параметрів сенсорів мобільного робота

Назва сенсера	Параметри Input	Параметри Output
Temperature Sensor DS18B20 [413]	-	Temperature
Magnetic encoder AS4050-ASSU [414]	-	Robot Position
IR Sensor E18-D90NK [415]	-	Obstacle
IR Sensor for Manipulator HOA 1404-002 [416]	-	Manipulator position
US sensor JSN-SR04T [417]	Start Pulse	

У відповідності з розробленою структурною схемою Sensor Noise, в основі якої закладено мікроконтролер STM32 [418], мікроконтролер дозволяє збирає показання з датчиків: температури, наявності та відстані до перешкод, магнітного енкодера позиціонування платформи. Датчики підключаються до портів мікроконтролера STM32 по шині GPIO. Оброблені дані пересилаються в основну систему (Control Node) за допомогою протоколу Modbus та шини RS 485.

На рисунку 9.9 показано розташування сенсорів на мобільному роботі. Для зручності розуміння розташування сенсорів, введемо такі скорочення:

OC1-OC4 – Obstacle Sensors;

OD1-OD2 – Distance to Obstracle Sensors;

tS1-tS2 – Temperature Sensors;

AS1 – Acceleration Sensor;

MS1-MS3 – Manipulator Possition Sensor;

ME1-ME4 – Magntic Sensor.

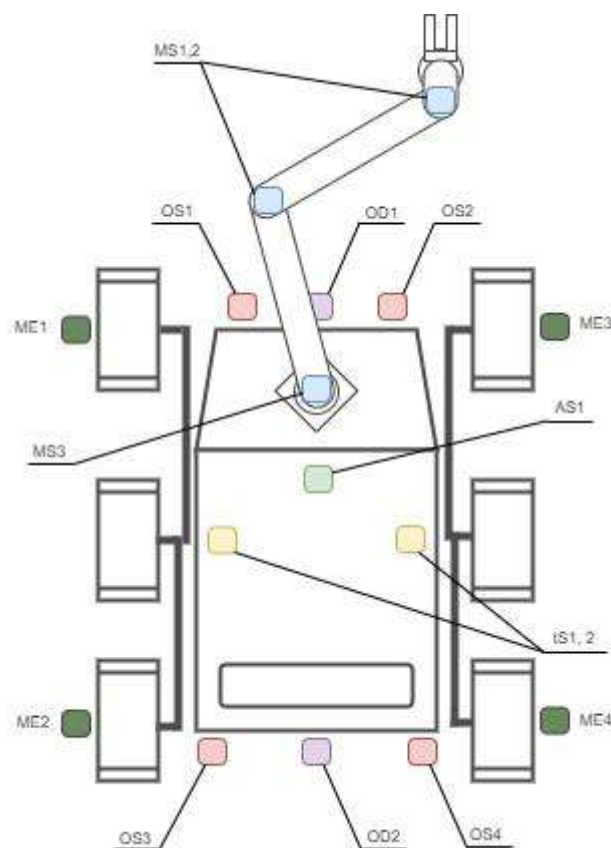


Рисунок 9.9 – Розташування сенсорів на мобільному маніпуляційному роботі

Магнітні енкодери AS4050 встановлені на колесах мобільного маніпуляційного робота [419]. Зчитування показників по енкодерах відбувається із частотою 20 Гц і передається через інтерфейс SPI у систему управління. Отримані дані з енкодерів постійно обробляються в основному циклі програми і передаються на мікроконтроллер UART1 у вигляді повідомлення, що передаються до Robot System через інтерфейс RS 485.

У розробленій конструкції мобільної робототехнічної маніпуляційної платформи використовуються наступні датчики визначення перешкод:

- інфрачервоні датчики E18 D80NK;
- ультразвукові датчики JSN_SR04T.

Крім того, використовуються датчики температури DS18B20, а також кінцеві датчики. Усі датчики зчитуються переважно в основному циклі з частотою 10 Гц. Датчики температури призначені для контролю температури

всередині корпусу робота і працюють за інтерфейсом 1Wire і видають температуру з точністю до десятих градусів.

Отримані з усіх датчиків дані формуються повідомлення і відправляються інтерфейсом UART2 мікроконтролера на запит від Robot System через інтерфейс RS 485.

У блоці Sensor Node розроблено свій протокол обміну повідомленнями та формат пакетів для передачі інформації між нею та Robot System. Формат пакета даних наведено у таблиці 9.3.

Таблиця 9.3 – Формат пакета даних у блоці Sensor Node

Field Name	Size of Field	Command Description
Address	1 Byte	The address of the device to which the packet is destined
Command	1 Byte	Package command. Specifies the content of the package data.
Size	1 Byte	The size of the packet data/ The maximum data size is 1460 bytes
Check Sum	2 Byte	Checksum of the entire preceding and following parts of the packet.
Parametr	N Byte	Package data. May vary in size for some commands

Основою системи управління є General Node є головний вузол системи управління, побудований на основі мікрокомп'ютера Raspberry PI 3 model B+. Цей вузол отримує всю необхідну інформацію за допомогою промислового протоколу Modbus. Як було розглянуто раніше, всі модулі мають можливість підключення до внутрішньої шини, що побудована на основі інтерфейсу RS 485.

Декілька окремих блоків таких як Navigation, Mapping Node та CNN Computer Vision Node підключаються до основної системи управління

мобільної робототехнічної платформи за допомогою спеціальних виділених інтерфейсів. Це пов'язано з тим, що дані блоки вирішують спеціалізовані завдання:

- локальної навігації та побудови локальної карти місцевості та оточення мобільного робота (блок Navigation and Mapping Node);
- розпізнавання та ідентифікації об'єктів на базі комп'ютерного зору, для вирішення спеціалізованих завдань, таких як розпізнавання, переміщення та знезараження небезпечних вибухових речовин (блок CNN Computer Vision Node).

На рисунку 9.10 наведено розроблену структуру взаємодії блоків: Navigation fтв Mapping Node and CNN Computer Vision Node з основною системою управління мобільним маніпуляційним роботом (Control Node).

Як можна бачити з рисунку 9.10, запропоновані такі рішення, які дозволять об'єднати через промисловий інтерфейс MODBUS блоки Navigation та CNN Computer Vision Node (блок Camera). Для реалізації блоку Camera, було віддано перевагу 3D-камері Intel RealSense D455 [*].

Основні характеристики 3D-камери Intel® RealSense™ D455 наведено в таблиці 9.4.

Таблиця 9.4 – Основні характеристики 3D-камера Intel® RealSense™ D455[*]

№	Параметр	Значення
1	Процесор	Intel® RealSense™ Vision Processor D4
2	Роздільна здатність стерео зображення	Up to 1280x720 active stereo depth resolution
3	Роздільна здатність RGB	Up to 1280x800 RGB resolution
4	Кут огляду	Diagonal Field of View over 90°
5	Відстань фокусування	Range 0.4m to over 10m (Varies with lighting conditions)

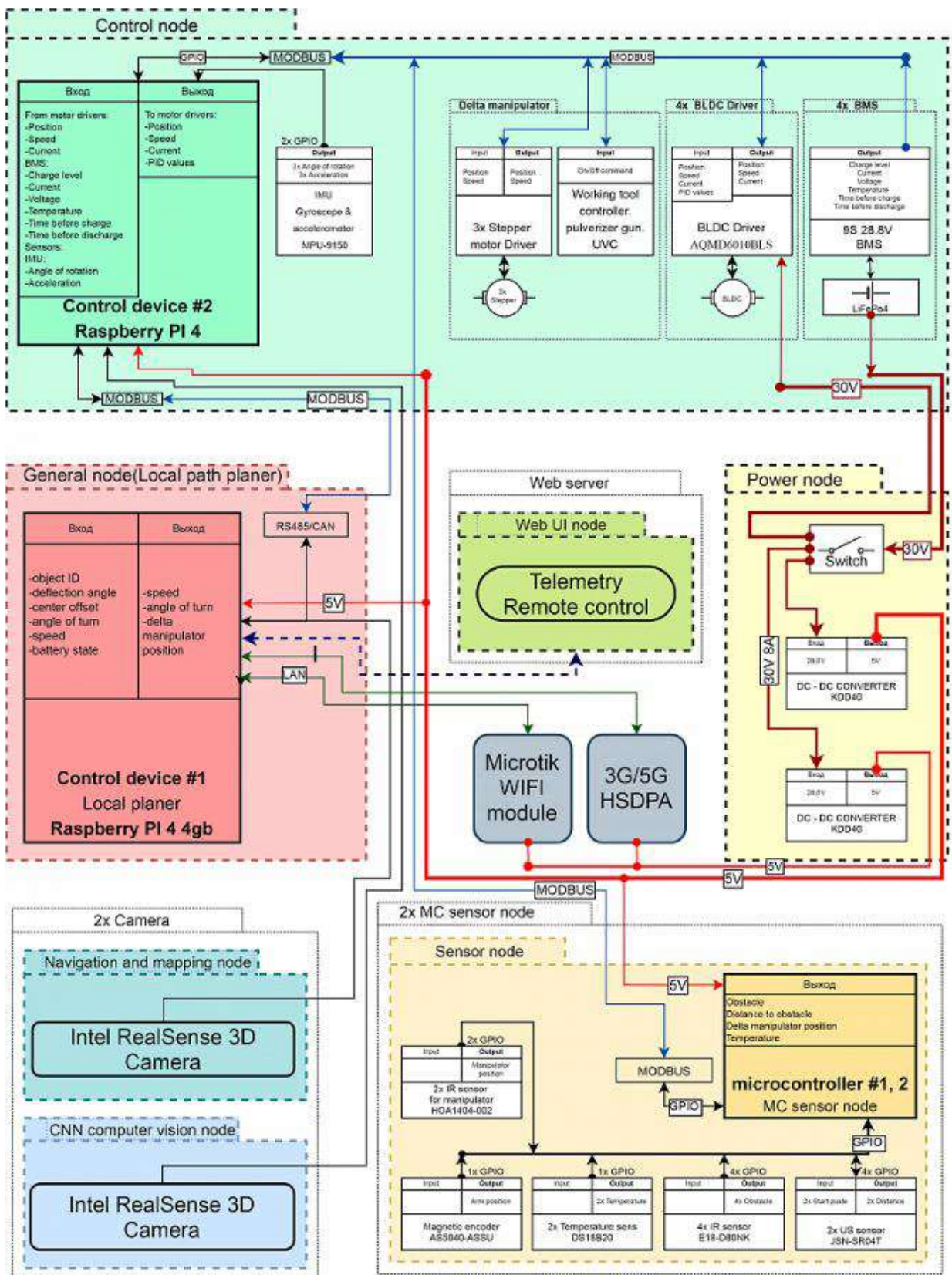
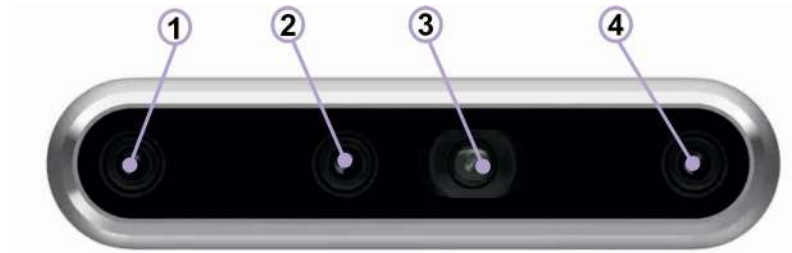


Рисунок 9.10 – Структура взаємодії блоків: Navigation, Mapping Node та CNN Computer Vision Node з основною системою управління мобільним маніпуляційним роботом

Загальний вигляд 3D-камера Intel RealSense D455 представлений на рисунку 9.11.



1,3 – датчик обсягу (ІЧ-датчик);

2 – ІЧ випромінювач;

3 – відеокамера

Рисунок 9.11 – Загальний вигляд 3D-камери Intel RealSense D455[420]

Intel RealSense Depth Camera D455 є на сьогодні топом лінійки камер глибини Intel RealSense Depth Camera D4xx. Вона оснащена інерційним вимірювальним модулем inertial measurement unit (IMU), який дозволяє визначити положення (рухи та повороти) у 6 ступенях свободи (6 degrees of freedom – 6DoF).

Вона комбінує дані з різних сенсорів з гіроскопами для визначення як обертання, так і руху за 3-ма осями, а також тангаж, пошук та захват і крен. Він застосовується для стабілізації зображення під час руху камери, що важливо в ігрових та вказівних пристроях. Особливістю даної моделі є те, що відстань між ІЧ-сенсорами збільшено до 95 мм шляхом розміщення їх по краях модуля, що покращує визначення глибини зображення (помилка не перевищує 2 % на дистанції 4 м) [421].

Navigation and Mapping Node (Рис. 9.10) вирішує задачу локальної навігації за допомогою алгоритму Visual Odometry [422]. Алгоритм заснований на використанні спеціальних точок на зображенні для визначення напрямку руху та пройденої дистанції. Функціональна діаграма реалізації Navigation and Mapping Node представлена рисунку 9.12.

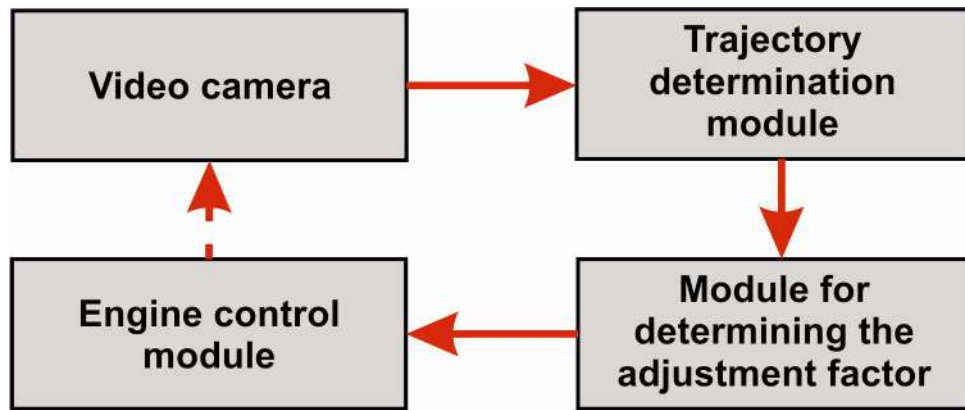


Рисунок 9.12 – Функціональна діаграма реалізації Navigation and Mapping Node

На основі даних візуальної одометрії головний модуль (Control Node) визначає кут зміщення платформи від траєкторії та обчислює коефіцієнт коригування швидкості двигуна приводу робота. Отримані коефіцієнти перетворюються на числові значення швидкості обертів двигунів. За допомогою Control Node відбувається керування двигунами постійного струму.

Вузол Web UI Node (блок Web server на рис. 9.10) призначений для виконання завдання віддаленого управління мобільною платформою оператором за допомогою Human-Machine Interface (HMI) [423].

Підключення до модуля управління відбувається за допомогою WiFi засобів зв'язку. Окремим модулем є Power Node. Даний модуль виконує завдання контролю напруги живлення, споживаного струму і визначення часу до наступного сеансу перезарядки. Для живлення двигунів використовується 30 В постійного струму, а живлення електронної частини – 5 В.

9.2 Розробка структури програмного забезпечення системи керування мобільним маніпуляційним роботом

Програмне забезпечення для управління платформою мобільного маніпуляційного робота працює в General Node на основі мікрокомп'ютера Raspberry PI 3 mobil B +. Ця програма призначена для прийняття команд від Server System та їх виконання.

Команди від серверного модуля передбачають передачу маршруту та завдань для виконання маніпулятором із спеціальним інструментом.

Отримавши маршрут, програма управління розбиває його на прямолінійні ділянки та зберігає його у виділеній ділянці пам'яті.

Далі завантажується локальний планувальник, який виконує завдання з прямолінійного руху та поворотів. У якості локального планувальника використовується пакет ROS Move_basic.

ROS Move_basic - это основной узел Navigation, он занимает доминирующее положение в задачах навигации, а некоторые другие пакеты являются его плагинами.

Для запуска move_base необходимо выбрать плагин, включающий три плагина: base_local_planner, base_global_planner и recovery_behavior, эти три плагина должны быть указаны, в противном случае система будет указывать значение по умолчанию. Navigation предоставляет нам множество плагинов-кандидатов, которые можно выбрать при настройке move_base.

ROS Move_basic – це основний вузол Navigation, він займає домінуюче положення в задачах навігації, а деякі інші пакети є його плагінами. Для запуску move_base необхідно обрати плагін, що включає три плагіни: base_local_planner, base_global_planner і recovery_behavior. Ці три плагіни повинні бути вказані, інакше система буде вказувати значення за замовчуванням. Navigation надає нам безліч плагінів-кандидатів, які можна обрати під час налаштування move_base.

Плагін base_local_planner:

- `base_local_planner`: реалізує два локальні алгоритми планування, розгортання траєкторії та DWA;
- `dwa_local_planner`: реалізований алгоритм локального планування DWA, який можна розглядати як покращену версію `base_local_planner`.

Плагін `base_global_planner`:

- `parrot_planner`: реалізує найпростіший алгоритм глобального планування;
- `navfn`: реалізовані алгоритми глобального планування Dijkstra та A*;
- `global_planner`: перевизначені алгоритми глобального планування Dijkstra та A*, які можна розглядати як покращену версію `navfn`.

Плагін `recovery_behavior`:

- `clear_costmap_recovery`: реалізує поведінку відновлення очищення картки витрат;
- `rotate_recovery`: реалізує поведінку відновлення обертання;
- `move_slow_and_clear`: досягається повільне відновлення поведінки.

Плагін можна обрати у налаштуванні параметра `move_base`. Оскільки параметрів занадто багато для налаштування, конфігурація зазвичай записується у файл `yaml`. Ми використовуємо `param/move_base_params.yaml` для збереження наведених вище параметрів.

На рисунку 9.13 показано Software structure schema програмної реалізації мобільного маніпуляційного робота.

Software structure schema (рис. 9.13) умовно поділена на вертикальні та горизонтальні рівні. Горизонтальний рівень (праворуч) показує рівні ROSCH framework, які складаються з:

- `sensors layer`;
- `preprocessed layer`;
- `perception layer`.

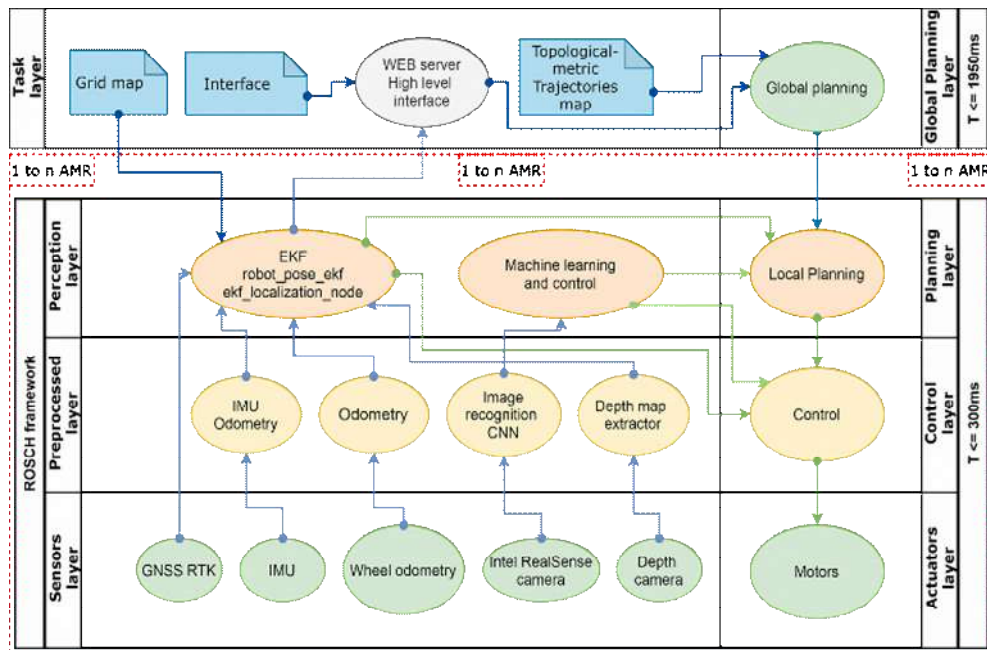


Рисунок 9.13 – Software structure schema

До рівня sensors layer входять:

- Global Navigation Satellite System Real Time Kinematic (GNSS RTK) – сукупність прийомів та методів отримання планових координат та висот точок місцевості сантиметрової точності за допомогою супутникової системи навігації за допомогою отримання поправок з базової станції, які приймає апаратура користувача під час зйомки;

- Inertial Measurement Unit (IMU) – це пристрій, який може вимірювати та повідомляти про питому вагу та кутову швидкість об'єкта, до якого він прикріплений. IMU зазвичай має у складі: гіроскопи: вимірювання кутової швидкості; акселерометри: вимірювання питомої сили/прискорення; магнітометри (додатково): вимірювання магнітного поля, що оточує систему.

- Wheel odometry – це процес визначення положення та орієнтації робота шляхом аналізу пов'язаних зображень камери. Він використовувався у широкому спектрі роботизованих додатків;

- Intel RealSense camera – сенсори розташовані на 3D-камері Intel RealSense D455 (рис. 9.11);

- Depth camera – модуль камери глибини на 3D-камері Intel RealSense D455 (рис. 9.11).

Рівень Preprocessed layer – призначений для попередньої обробки сигналу, що надходить із Sensors layer і містить наступну структуру:

- IMU odometry;
- odometry;
- image recodnition CNN;
- depth map rxtraction.

Perception layer – програмний рівень призначений для розуміння інформації, що надходить з Preprocessed layer. Він складається з наступних блоків:

- EKF robot_pose_ekf и ekf_localization_node;
- machine learning and control.

На верхньому рівні знаходиться Task layer, який складається з наступних блоків:

- grid map – блок побудови карти місцевості та розподілення його на сітку;
- interface – блок реалізації НМІ локального керування;
- Web server – Web server та верхній рівень реалізації НМІ;
- Topological metric – блок обробки топологічних міток;
- Trajectoris map – блок побудови траєкторії переміщення мобільного маніпуляційного робота за інформацією з блоку grid map.

З наведеного на рисунку 9.13 можна побачити, що програмна частина складається з двох основних шарів: ROSCH framework і шар Task Layer [Rosch]. У свою чергу, ROSCH framework представлений трьома шарами: Sensor Layer, Preprocessed Layer, Perception Layer.

У шарі Task Layer виконуються завдання побудови карти місцевості, прокладання шляхів перешкод, організації взаємодії з оператором через WEB-інтерфейс.

Модуль move_basic використовується як локальний планувальник із частотою опитування 50 Гц. Вузол move_basic виконує завдання простої

навігації. Планування шляху складається з повороту по відношенню до цілі та подальшого руху у напрямку до неї.

Вхідними параметрами є `/move_base/goal`, `/sonars` (датчики УЗ та ІЧ) та `/TF` (tf між `map` і `base_link` використовується для планування, якщо він доступний. Інакше використовується tf між `odom` та `base_link`. Tf між `odom` та `base_link` завжди використовується для виконання.)

Фільтр Калмана використовується для об'єднання даних про місцезнаходження з 3 джерел: одометрії з коліс (драйвера двигунів), одометрії з енкодерів та IMU датчика.

Після фільтра Калмана поточна позиція передається до `move_basic` для подальшого орієнтування.

Вихідні параметри – це `/cmd_vel`, який являє собою лінійну та кутову швидкість для робота. Після цього дані про задану швидкість використовуються пакетом `/robot_hardware_controller`, який розбиває задані швидкості на 2 швидкості для кожного колеса (`/right_wheel/setpoint` і `/left_wheel/setpoint`).

Далі `/cmd_vel` із заданими швидкостями для кожного колеса використовуються в пакеті ПД-регулятора, який дає на вихід скориговану швидкість у вигляді значення величини ШІМ-сигналу для двигунів коліс. (топіки `/right_wheel/controll_effort` та `/left_wheel/controll_effort`).

За допомогою скрипта `driver_connection` йде спілкування з драйверами управління двигунами коліс за протоколом MODBUS, а саме зчитування положення та поточної позиції та завдання поточної швидкості. Список параметрів `Move_basic` представлений у таблиці 9.5.

Таблиця 9.5 – Список параметрів Move_basic

Параметр	Діапазон / Поточне значення	Одини- ці	Коментар
1	2	3	4
min_angular_velocity	0 - 2 / 0.05	r/s	
max_angular_velocity	0 - 2 / 1.2	r/s	
angular_acceleration	0 - 0.5 / 0.05	r/s^2	
angular_tolerance	0 - 0.1 / 0.003	r	
min_linear_velocity	0 - 2 / 0.02	m/s	
max_linear_velocity	0 - 2 / 1.5	m/s	
linear_acceleration	0 - 1 / 0.4	m/s^2	
linear_tolerance	0 - 0.1 / 0.002	m	
min_side_dist	0.1	m	Мінімальна відстань з кожного боку
max_lateral_deviation	3	r	Максимальне відхилення від лінійного шляху до переривання
max_angular_deviation	0.017 (1 deg)	r	Максимально допустиме відхилення від прямого шляху
max_lateral_rotation	0.02	r / s	Максимальна кутова швидкість на лінійній ділянці
side_recover_weight	0 - 1 / 0.005	-	Коефіцієнт для відновлення після об'їзду бічних перешкод

Продовження таблиці 9.5

1	2	3	4
obstacle_wait_limit	0 - 60 / 10	s	Час очікування, поки перешкода зникне
reverse_without_turning_threshold	0.5	m	Відстань для руху назад, на яких обертання не виконуватиметься
robot_width	0.57	m	
preferred_planning_frame	odom	-	
preferred_driving_frame	odom	-	
lateral_kp	1.0	-	Кутовий ПІД-регулятор для коригування бокового положення робота
lateral_ki	0.02	-	
lateral_kd	1.0	-	

Для зручності розуміння організації програмних модулів був розроблений вузловий граф Robot System (системи управління) який представлений на рисунку 9.14.

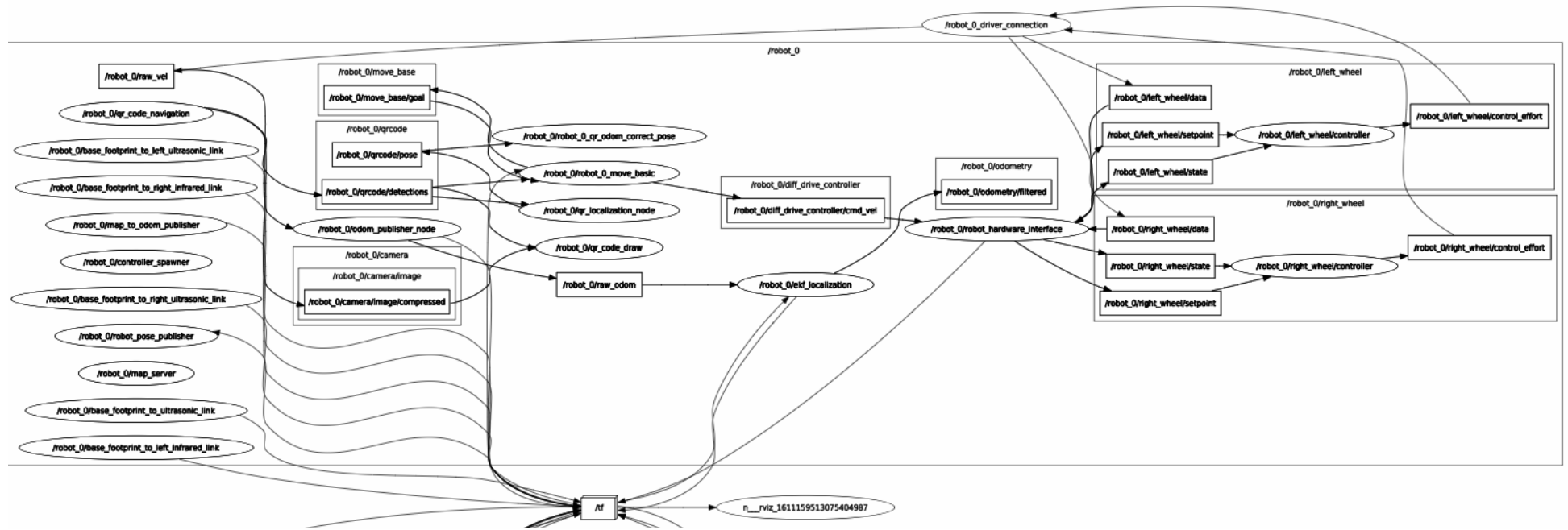


Рисунок 9.13 – Вузловый граф Robot System

Вхідні параметри: `/move_base/goal`, `/sonars` (ультразвукові та інфрачервоні датчики) та `/tf` (відстеження кількох координатних кадрів між `map` та `base_link`, використовується для планування маршруту). Якщо трансформацію між `map` і `base_link` визначити немає можливості, використовується TF між `odom` та `base_link`. TF між `odom` та `base_link` завжди використовується для виконання.

Для отримання траєкторії руху використовуються дані з трьох джерел: одометр із коліс (дані з драйвера двигунів), одометр з енкодерів та датчик IMU.

Одометр з коліс (`Odom0`) використовує дані про положення робота по осях X і Y , а також орієнтацію по осі Z . За допомогою `driver_connection` зчитуємо поточну швидкість коліс та відправляємо в `topic/raw_vel`. Далі скрипт `odom_publisher` перетворює цю швидкість на позицію і відправляє її в `topic/raw_odom`, що використовується в EKF.

Одометр з енкодерів (`Odom1`) використовує дані про положення робота по осях X і Y , а також орієнтацію по осі Z . Обробляється аналогічно одометру коліс, тільки використовуються дані з енкодерів (скрипт `encoders_receive.py`), а кінцева публікація відбувається в топіку `/enc_odom`.

Дані з датчика IMU MPU9250 (`Imu0`) надають інформацію про швидкість роботи по осях X та Y , а також кутову швидкість осі Z .

Поєднання інформації із трьох джерел проводиться із застосуванням фільтра Калмана. Таким чином визначається точне місцезнаходження мобільної платформи. У скрипті `imu_publisher.py` отримуємо дані з датчика (акселерометра, гіроскопа та магнетометра), обробляємо їх та публікуємо в топіку `/imu/data_raw` (тільки гіроскоп та акселерометр). Дані магнетометру і температура публікуються в топіку `/imu/mag` і `/imu/temperature` відповідно). Після цього використовується фільтр Маджвіка (`imu_filter_madgwick`) [424] і отримані дані заносяться в топік `/imu/data`.

У системі використовується пакет `robot_localization` для використання функціональності EKF. Частота, з якою фільтр виводить оцінку становища –

50 Гц. Ekf_localization_node використовує тривимірну модель всеспрямованого руху. Але за допомогою параметра "two_d_mode: true" жодна інформація про рух вгору-вниз не використовуватиметься при оцінці поточного положення, що необхідно при рухах на поверхні. Параметр sensor_timeout: 0,01 відповідає за час, через який фільтр розпочне прогноз нової позиції за відсутності нових даних від зазначених джерел.

Принцип поєднання даних із різних джерел поданий на рис. 9.14.

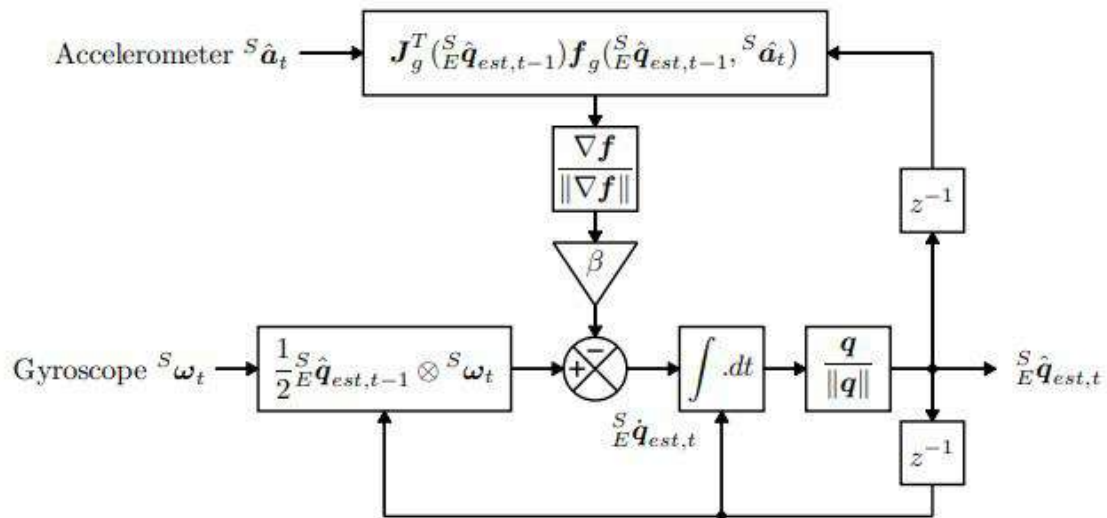


Рисунок 9.14 – Подання блок-схеми повного фільтра орієнтації для реалізації IMU

Після застосування фільтра Калмана поточна позиція передається на модуль move_basic для подальшої орієнтування. Вихідними параметрами даного блоку є топик /cmd_vel, що представляє собою лінійну і кутову швидкість для роботи.

Наступним етапом є визначення швидкості кожної пари коліс (рис. 9.13). Для цього дані про задану швидкість надаються пакету /robot_hardware_controller, який розбиває задану швидкість на дві складові – для кожної пари коліс (/right_wheel/setpoint та /left_wheel/setpoint).

Далі топіки із заданими швидкостями для кожного колеса використовуються в пакеті ПД-регулятора, який дає на вихід скориговану швидкість у вигляді значення величини сигналу ШІМ для двигунів коліс.

За допомогою скрипта `driver_connection` йде спілкування з драйверами управління двигунами коліс за протоколом MODBUS, а саме зчитування положення та поточної позиції та завдання поточної швидкості.

9.3 Метод визначення траєкторії руху у невизначеному просторі

Суть розроблюваного методу полягає у поєднанні методу пошуку оптимальної траєкторії руху, коли всі перешкоди визначені, та методу візуальної одометрії для визначення змін в навколишньому просторі [425].

Синтезуючи ці два методи, ми отримуємо можливість постійного пошуку та аналізу розташування нових перешкод, що з'являються у навколишньому просторі та швидкого реагування на знайдені зміни у реальному часі.

Перед адаптивною візуальною системою ставляться такі завдання: визначення розташування робота у вихідній позиції та при переміщенні; орієнтація робота у просторі щодо початкової точки; визначення та стеження за об'єктами, що перебувають у робочому просторі; виділення непрохідних ділянок.

Для аналізу оточуючого середовища використовуватиметься метод візуальної одометрії, суть якого викладено у попередньому підрозділі. Метод дозволяє прив'язати перешкоди до локальної карти місцевості. Виявлені зміни відразу наносяться на карту і надалі використовуються для оптимального прокладання маршруту. Прокладання маршруту за даними карти відбувається за допомогою алгоритму A^* .

Таким чином, ми маємо три незалежні процеси: пошук перешкод, модернізація карти та прокладання маршруту. Метод, що розробляється, схожий з відомим принципом переміщення роботів у просторі, який

називається SLAM (метод одночасної навігації та побудови карти). Відмінність полягає в тому, що в традиційному методі робот не має повного уявлення про навколишній простір і перш за все проводить дослідження простору та побудову карти.

У цьому випадку ми припускаємо, що картка вже відома і завантажена в пам'ять мобільного пристрою. Наприклад, це дуже часто буває, коли робот працює на виробничій ділянці, де заздалегідь відоме розташування обладнання.

Грунтуючись на цьому припущення, можна з упевненістю сказати, що на першому етапі відбувається швидке прокладання шляху від початкової точки до заданої цілі. У процесі переміщення робот постійно сканує простір та слідкує за змінами в навколишньому просторі. Зміни можуть бути спричинені появою людини перед мобільним пристроєм, появою іншого мобільного робота або переміщенням продукції.

Отже, визначивши місце нової перешкоди або її зникнення, модуль управління роботом перебудовує траєкторію руху і визначає оптимальний маршрут для даної поточної ситуації. Посівність цих етапів покладена в модель адаптивної системи управління мобільною платформою в невизначеному середовищі, яка представлена на рисунку 9.15.

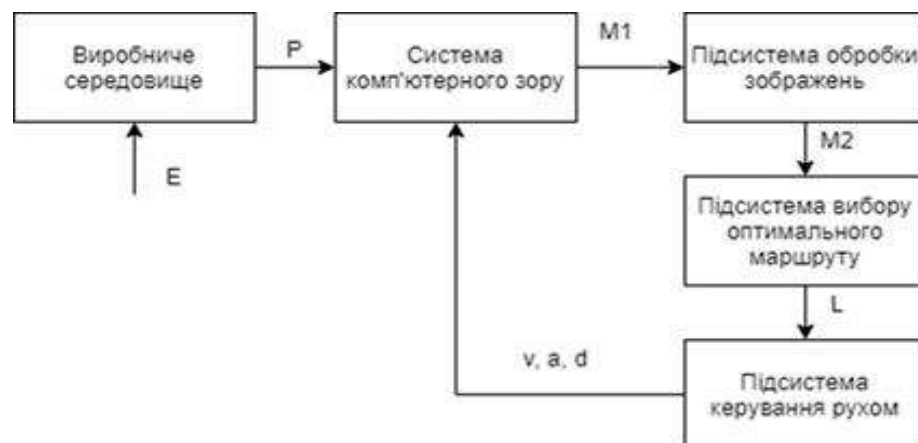


Рисунок 9.15 – Модель адаптивної системи керування мобільною платформою

Модель адаптивної системи управління мобільною платформою, можна описати як безліч яка складається з: навколишнього середовища яке складається з об'єктів, що не рухаються і є елементами обладнання (O) або конструкцій приміщення, а також об'єктів, що рухаються, таких, як людина, або інші мобільні роботи (U).

Таким чином, будь-яка точка простору може бути представлена у вигляді:

$$P^* \{[O_1, O_2, \dots, O_n], [U_1, U_2, \dots, U_n]\} \quad (9.1)$$

де P^* – координати опорної точки об'єктів у приміщенні;

n – кількість опорних точок у просторі.

Навколишнє середовище можна описати набором опорних точок, що складають просторовий фон, що заважає, або не заважає руху:

$$P = \{P_1^*, P_2^*, \dots, P_n^*\} \quad (9.2)$$

Для цього представимо M , як модель переміщення:

$$M_1 = \{S, W\} \quad (9.3)$$

де S – метод визначення візуальних координат;

W – параметри методу.

$$L = \{\Delta X, \Delta d\} \quad (9.4)$$

де X – локалізація мобільної платформи;

d – напрямок руху платформи.

На виході підсистеми управління рухом отримуємо параметри на рух мобільної платформи, а також і на зсув системи комп'ютерного зору, яка жорстко пов'язана з платформою (v, a, d) . Параметр визначає зміну швидкості руху платформи. Параметр a визначає кутове зміщення, а параметр d – напрямок руху.

9.4 Розробка програмного модуля для моделювання переміщення мобільного робота

Виходячи з проведеного аналізу, програмний модуль для моделювання методу пошуку оптимального шляху переміщення мобільної платформи серед невизначених перешкод має такі незалежні процеси:

а) TraceRoute, що використовується для побудови траєкторії руху на основі даних про знайдені перешкоди;

б) TimerProces – незалежний складний процес, що запускає такі функції:

- сканування простору та пошук перешкод;
- візуалізація ефекту роботи сканера навколишнього простору;
- розбудова віртуальної карти;

в) переміщення цифрового двійника на віртуальній карті виробничого середовища знайденим шляхом.

Структурна схема програмного модуля для моделювання методу пошуку оптимального шляху переміщення мобільної платформи серед невизначених перешкод, представлена рисунку 9.16.

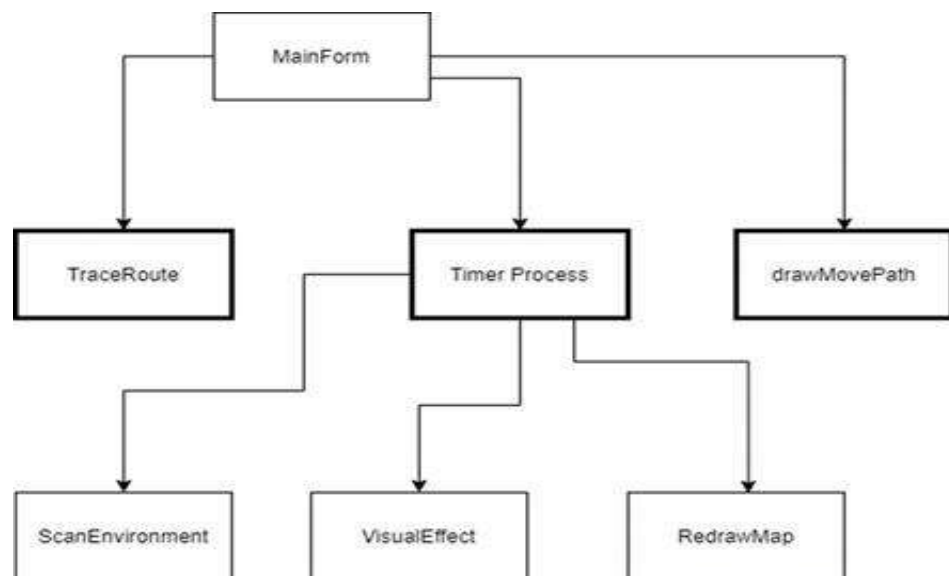


Рисунок 9.16 – Структурна схема програми для моделювання методу пошуку оптимального шляху переміщення мобільної платформи серед невизначених перешкод

Для візуалізації процесу сканування простору використовується функція `drawScan()`. Для поєднання всіх процесів розроблено функцію підрахунку модельного часу `timer`.

Після завершення переміщення мобільної платформи та досягнення кінцевої точки, таймер вимикається та викликається функція `drawMovePath` для візуалізації пройденого шляху. Ця функція викликається після досягнення мобільної платформи кінцевої точки. Вона малює на карті пройдений шлях роботом для візуалізації всіх зсувів у процесі пошуку траєкторії.

Шлях будується за даними, що зберігаються у масиві `MovePath`. У цей масив заносяться всі точки, в яких знаходився мобільний робот у процесі руху. Масив `MovePath` змінює функцію зміщення мобільної платформи. На рисунку 9.17 показаний приклад роботи алгоритму на перших кроках роботи та результат прокладання шляху.

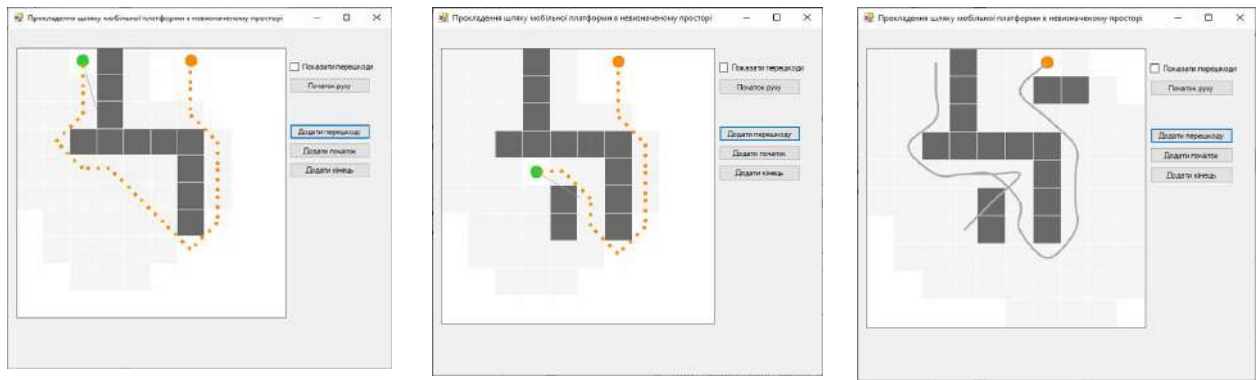


Рисунок 9.17 – Приклад роботи алгоритму на перших кроках роботи та результат прокладання шляху

Шлях може проходити через наявні перешкоди тому, що на момент попереднього проходу їх ще не було, а вони з'явилися при зміні траєкторії руху і були враховані під час прокладання нового шляху.

9.5 Розробка програмного модуля для керування маніпуляційним маніпулятором мобільного робота

Грунтуючись на запропонованій структурі мобільного маніпуляційного робота (рис. 9.9) було складено ескіз маніпулятора, представлений на рисунку 9.17. Плече 1 має довжину 380 мм, а плече 2 – 200 мм.

Як серводвигун 1 використовується кроковий двигун, як серводвигун 2 використовується двигун постійного струму з редуктором. Захватом управляє серводвигун із вбудованим контролером управління.

Усі двигуни управляються за допомогою спеціальних модулів драйверів відповідного типу двигунів. Програма, що розробляється, повинна мати можливість зв'язуватися з головним модулем управління за допомогою будь-якого інтерфейсу і передавати спеціально розроблені для цього команди для включення або вимкнення певного двигуна.

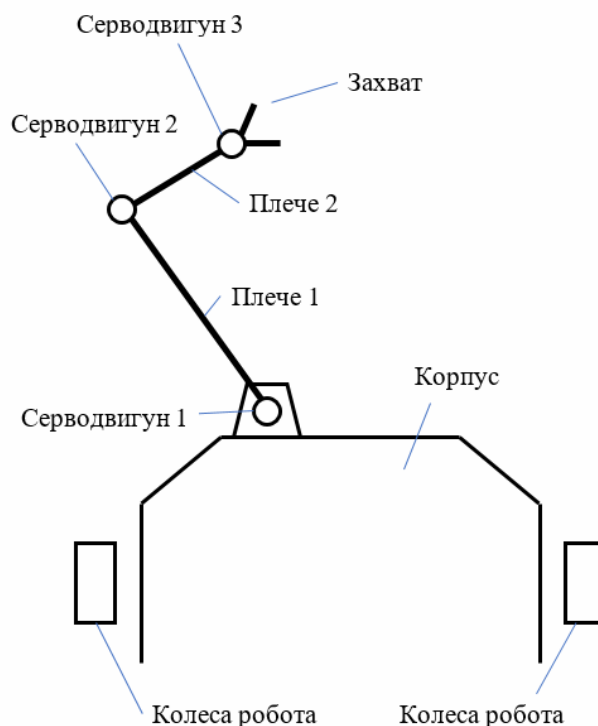


Рисунок 9.17 – Ескіз маніпулятора мобільного робота

Особливістю даного типу маніпулятора є відсутність зворотного зв'язку з модулем управління, що реалізується. Це вимагає певних вимог до програмного модуля керування. Одним із рішень є використання візуального контакту з маніпулятором робота під час керування. Тобто оператор повинен мати можливість за допомогою відеокамери або наочно контролювати процес переміщення маніпулятора.

Іншим варіантом є визначення кута повороту плеча за допомогою часу, що минув після отримання команди керування. Якщо знати час, що витрачається весь шлях, можна з певною похибкою підрахувати необхідний проміжок часу для зміщення механізму на заданий кут. Головне вікно програми керування маніпулятором наведено на рисунку 9.18.

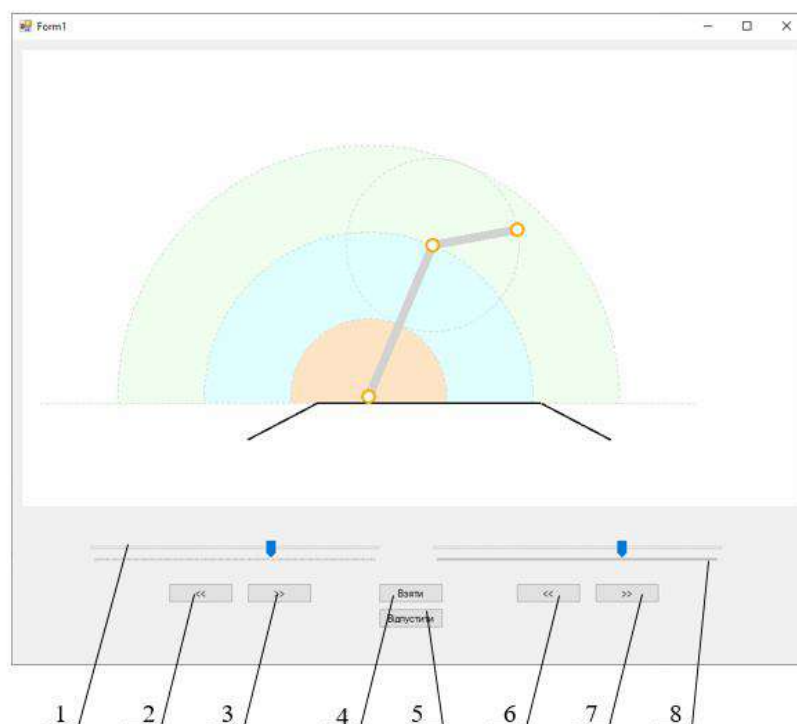


Рисунок 9.18 – Головне вікно програми управління маніпулятором

У ході розробки НМІ було виділено дві області:

- область управління;
- область відображення поточного стану маніпулятора.

На рисунку 9.18 позначені цифрами наступні графічні елементи інтерфейсу:

- 1 – елемент управління великим плечем маніпулятора;
- 2 – кнопка «Вліво» для керування великим плечем маніпулятора;
- 3 – кнопка «Вправо» для керування великим плечем маніпулятора;
- 4 – кнопка «Взяти» для керування схватом;
- 5 – кнопка «Відпустити» для керування схватом;
- 6 – кнопка «Вліво» для керування коротким плечем маніпулятора.

В інтерфейсі програми використовуються такі візуальні компоненти, як кнопки та слайдери. Слайдери 1 та 8 призначені для точного позиціонування маніпулятора комп'ютерною мишею. Слайдер 1 управляє великим плечем маніпулятора, слайдер 2 – керувати маленьким плечем.

При переміщенні слайдера відбувається перебудова стану маніпулятора на екрані ПК. З огляду на відсутність енкодерів на двигунах маніпулятора зображення на екрані не завжди відповідає поточному стану маніпулятора на роботі. Тому для точного управління використовуються кнопки 2-7.

Для обробки стану слайдера використовується обробник події ValueChanged. Приклад функції обробника показаний нижче:

```
private void trackBar6_ValueChanged(object sender, EventArgs e)
{
    TrackBar tr=(TrackBar)sender;
    angle2 = 360 – tr.Value;
    panel1.Refresh();
}
```

Цей код обчислює кут обороту невеликого плеча маніпулятора і надає його змінній angle2. Аналогічна функція використовується і для першого слайдера, вона лише обчислює кут зсуву великого плеча маніпулятора та привласнює йому змінну angle1.

При старті програми задаються початкові значення слайдерів:

```
private void Form1_Load(object sender, EventArgs e)
{
    trackBar5.Maximum=180;
    trackBar5.Value = angle1;
    trackBar6.Minimum=-180;
    trackBar6.Maximum=180;
    trackBar6.Value = angle2;
}
```

Для обробки стану кнопок використовуються події: `button_MouseDown` та `button_MouseUp`. Перша подія запускає таймер 1, а друга вимикає його. Перша кнопка працює з таймером 1, друга з таймером 2. Кнопка «<<» для керування маленьким плечем маніпулятора працює з таймером 3, кнопка «>>» для керування коротким плечем маніпулятора – з таймером 4.

На рис. 9.19 наведено фрагмент НМІ програми керування, що відображає поточний стан маніпулятора.

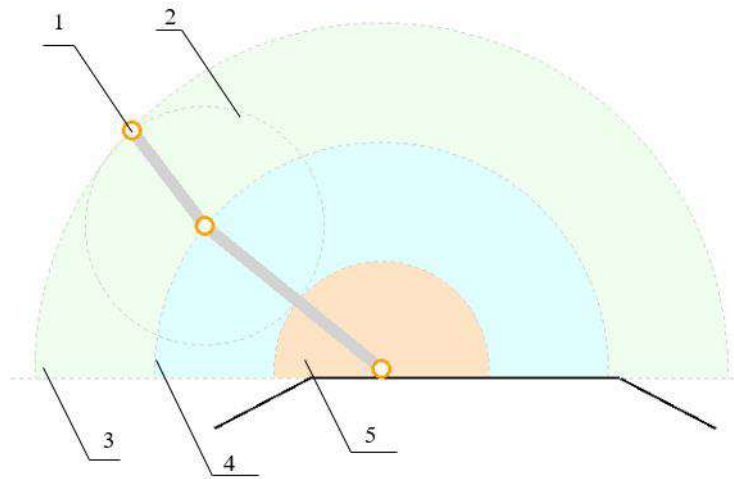


Рисунок 9.19 – Фрагмент НМІ програми управління, що відображає поточний стан маніпулятора

Основні елементи програми управління НМІ: 1 - положення захвату; 2 – траєкторія руху захвату маніпулятора; 3 – максимально можлива траєкторія руху; 4 – траєкторія руху середнього суглоба маніпулятора; 5 – зона недосяжності маніпулятора.

Розрахунки позиції суглобів маніпулятора представлено на рис. 9.20. Як видно, перший суглоб $L1$ закріплений на основі і повернутий на кут $Q1$. Другий суглоб $L2$ кріпиться до кінця першого суглоба і повернутий щодо нього на кут $Q2$. Робочий орган (захват) маніпулятора знаходиться наприкінці другого суглоба.

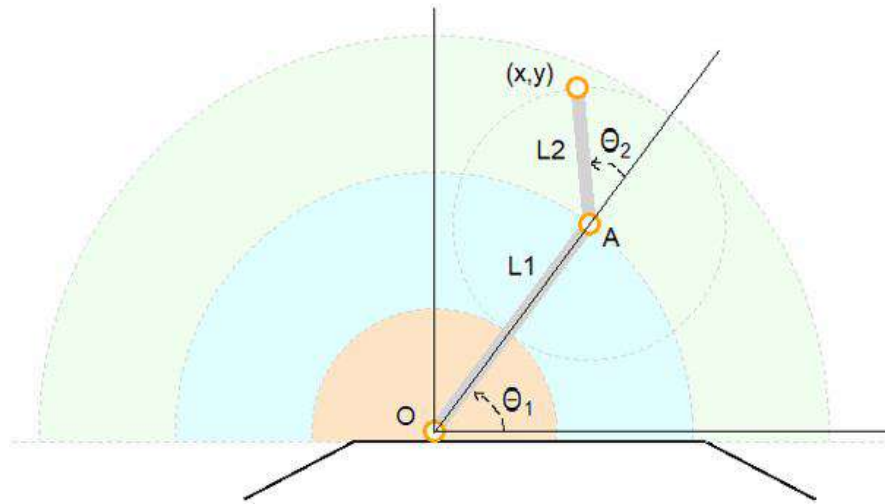


Рисунок 9.20 – Вихідні дані для розрахунку позиції суглобів маніпулятора

Пряме завдання кінематики полягає у знаходженні координат робочого органу (x, y) по заданим $L1$, $L2$, $Q1$, $Q2$ – це відповідно довжини плеча та ліктя маніпулятора, які в даному випадку визначені конструкцією маніпулятора.

Для вирішення цього завдання маємо дві системи відліку – перша, пов'язана з точкою кріплення плеча $L1$ – O , а друга – з початком координат у точці кріплення ліктя – A . Для графічного зображення маніпулятора знайдемо зміщення малого плеча щодо великого:

$$XA = L1 \cdot \cos(Q1) \quad (9.5)$$

$$YA = L1 \cdot \sin(Q1) \quad (9.6)$$

Координати розташування схвату (x, y) у системі відліку ліктя приймемо:

$$x' = L2 \cdot \cos(Q2) \quad (9.7)$$

$$y' = L2 \cdot \sin(Q2) \quad (9.8)$$

З рисунку 9.20 можна бачити, що локати повернуто щодо плеча на кут $Q1 + Q2$, отже:

$$x' = L2 \cdot \cos(Q1 + Q2) \quad (9.9)$$

$$y' = L2 \cdot \sin(Q1 + Q2) \quad (9.10)$$

Таким чином, координати схвату (x, y) можна обчислити так:

$$x = XA + x = L1 \cdot \cos(Q1) + L2 \cdot \cos(Q1 + Q2) \quad (9.11)$$

$$y = YA + y = L1 \cdot \sin(Q1) + L2 \cdot \sin(Q1 + Q2) \quad (9.12)$$

Виходячи з цього, позиція кожного плеча обчислюється окремо за допомогою наступної функції:

```
private void DreawLine(Graphics g)
{
    Point plecho_point1 = new Point(plecho1_start_x, plecho1_start_y);

    plecho_Middle = new Point((int)(plecho1_start_x - plecho1 *
Math.Cos(angle1 * Math.PI / 180)),
(int)(plecho1_start_y - plecho1 * Math.Sin(angle1 * Math.PI /
180)));

    Point plecho_point2 = new Point((int)(plecho_Middle.X - plecho2 *
Math.Cos((angle1 - angle2) * Math.PI / 180)),
(int)(plecho_Middle.Y - plecho2 * Math.Sin((angle1 - angle2) *
Math.PI / 180)));

    plecho_Start = plecho_point1;
    plecho_End = plecho_point2;
    DrawNormal(g, plecho_Start, plecho_End, plecho_Middle, 25);
}
```


}

Таким чином розроблений модуль НМІ управління для візуалізації переміщення ланок маніпулятора в просторі. Цей модуль взаємодіє із системою управління маніпулятором через послідовний провідний або бездротовий інтерфейс.

Команди відправляються послідовно. Кожна команда впливає на роботу конкретного двигуна залежно від її типу. Головний НМІ програми поєднує візуальні компоненти та обробляє події від активних компонентів.

9.6 Моделювання та експериментальні дослідження

У процесі руху мобільного маніпуляційного робота виникає ефект ковзання коліс, внаслідок чого дані одометра можуть не співпадати з реальним станом підсистеми планування траєкторії руху. На основі даних з інших датчиків (наприклад, візуальної одометрії) виконується корекція: скидається Y-компонента координат одометра. Нові дані беруться з модуля візуальної одометрії або інших працюючих у поточному завданні (наприклад, дані QR-коду).

Таким чином, робот отримує свою позицію та виправляє її за допомогою поворотів. Приклад моделювання коригування одометрії Y-компонента можна побачити на рисунку 9.21 (червона лінія на 23,5 с та 25,5 с).

Для роботи блоку регулятора PID використовуються такі параметри: `lateral_PID`, `max_lateral_rotation`, `max_angular_deviation`, `max_lateral_deviation` для використання цієї корекції.

Відповідні значення коефіцієнтів (`lateral_kp=1.0`, `lateral_ki=0.02`, `lateral_kd=1.0`). Ця поправка впливає на кутову швидкість у процесі лінійного руху. Показано на рис. 9.21 світло-синьою лінією. Червона лінія є зміщенням робота по осі Y від нуля.

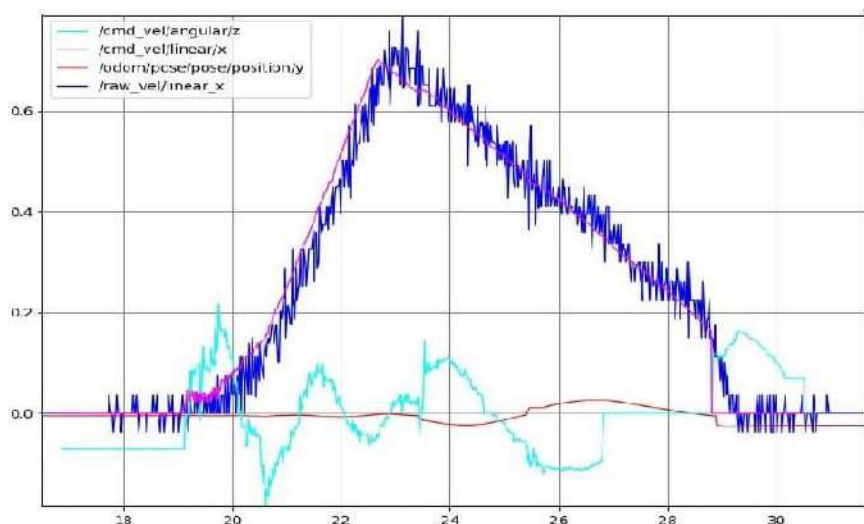


Рисунок 9.21 – Приклад експериментального моделювання коригування одометрії

Приклад контролю лінійної та кутової швидкості за допомогою сенсорів показаний на рисунку 9.22. Тут показані лінійна та кутова швидкості для випадку, коли мобільна платформа обертається на місці на 360° . При цьому швидкість обертання спочатку збільшується до максимального значення, а потім зменшується до повної зупинки.

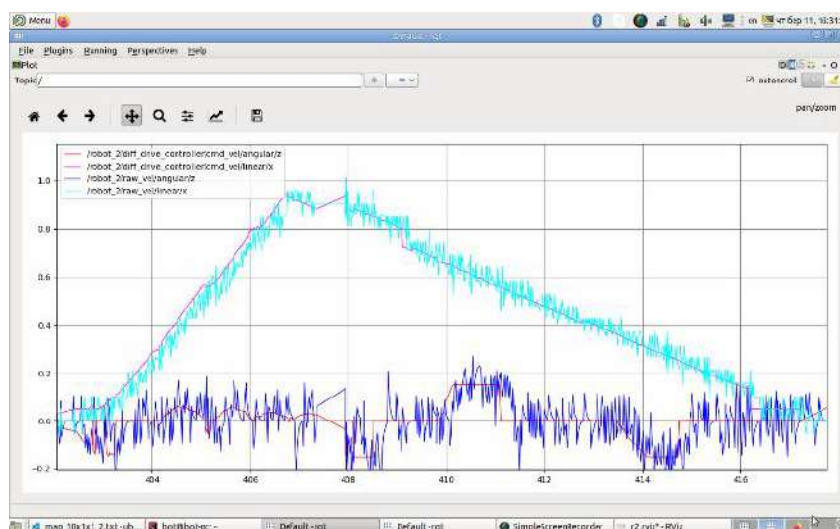


Рисунок 9.22 – Лінійна та кутова швидкості тестового зірка маніпуляційного робота

ПЕРЕЛІК ПОСИЛАНЬ

1. Chengchao Bai; Jifeng Guo; Hongxing Zheng. (2019). Three-Dimensional Vibration-Based Terrain Classification for Mobile Robots. *IEEE Access*. Volume: 7. Page(s): 63485 – 63492. DOI: 10.1109/ACCESS.2019.2916480
2. Lee, H., Chung, W. (2018). Terrain Classification for Mobile Robots on the Basis of Support Vector Data Description. *Int. J. Precis. Eng. Manuf.* 19, P.1305–1315. DOI:10.1007/s12541-018-0154-4
3. Prabin Kumar Panigrahi, Sukant Kishoro Bisoy. (2021). Localization strategies for autonomous mobile robots: A review. *Journal of King Saud University - Computer and Information Sciences*. DOI: 10.1016/j.jksuci.2021.02.015
4. Hyunsuk Lee; Woojin Chung. (2021). A Self-Training Approach-Based Traversability Analysis for Mobile Robots in Urban Environments. *The IEEE International Conference on Robotics and Automation (ICRA)*. 30 May-5 June. Xi'an, China. DOI: 10.1109/ICRA48506.2021.9561394.
5. Francisco Rubio, Francisco Valero and Carlos Llopis-Albert. (2019). A review of mobile robots: Concepts, methods, theoretical framework, and applications. *International Journal of Advanced Robotic Systems*. March-April. P 1–22. DOI: 10.1177/1729881419839596.
6. Hamid Taheri, Chun Xia Zhao. (2020). Omnidirectional mobile robots, mechanisms and navigation approaches. *Mechanism and Machine Theory*. Volume 153, November 2020. DOI: 10.1016/j.mechmachtheory.2020.103958.
7. Seyed Mojtaba Shafaei, Hossein Mousazadeh. (2021). Development of a mobile robot for safe mechanical evacuation of hazardous bulk materials in industrial confined spaces. *Journal of Field Robotics*. First published: 16 November 2021. DOI: 10.1002/rob.22047.

8. Yanan Liu; Rui Fan; Bin Yu; M. Junaid Bocus; Ming Liu; Hepeng Ni; Jiahe Fan; Shixin Mao. (2018). Mobile Robot Localisation and Navigation Using LEGO NXT and Ultrasonic Sensor. *The IEEE International Conference on Robotics and Biomimetics (ROBIO)*. 12-15 Dec. Kuala Lumpur, Malaysia. DOI: 10.1109/ROBIO.2018.8665350.
9. Kim, Changwon, Junho Suh, and Je-Heon Han. (2020). Development of a Hybrid Path Planning Algorithm and a Bio-Inspired Control for an Omni-Wheel Mobile Robot. *Sensors*. 20, no. 15: 4258. DOI:10.3390/s20154258.
10. Mohammed Rafeeq; Siti Fauziah Toha; Salmiah Ahmad; Mohd Asyraf Razib. (2020). Locomotion Strategies for Amphibious Robots-A Review. *IEEE Access*. Volume: 9. Page(s): 26323 – 26342. DOI: 10.1109/ACCESS.2021.3057406.
11. Sebastián Roa Prada. (2022). Design and Construction of a Spherical Mobile Robot Prototype for Monitoring Oil Palm Plantations. *ASME International Mechanical Engineering Congress and Exposition*. Paper No: IMECE2021-67327, V07BT07A011. DOI: 10.1115/IMECE2021-67327
12. Luis A. Mateos; Wei Wang; Banti Gheneti; Fabio Duarte; Carlo Ratti; Daniela Rus. (2019). Autonomous Latching System for Robotic Boats. *The International Conference on Robotics and Automation (ICRA)*. 20-24 May. Montreal, QC, Canada. DOI: 10.1109/ICRA.2019.8793525.
13. Bernd Henze; Ribin Balachandran; Máximo A. Roa-Garzón; Christian Ott; Alin Albu-Schäffer. (2018). Passivity Analysis and Control of Humanoid Robots on Movable Ground. *The IEEE Robotics and Automation Letters*. Volume: 3, Issue: 4. Page(s): 3457 – 3464. DOI: 10.1109/LRA.2018.2853266.
14. Pons, P., Jaen, J. (2020). Interactive spaces for children: gesture elicitation for controlling ground mini-robots. *J Ambient Intell Human Comput*. Volume: 11. P. 2467–2488. DOI:10.1007/s12652-019-01290-6.

15. Deyby Huamanchahua; Danitza Yalli-Villa; Angel Bello-Merlo; Jalber Macuri-Vasquez. (2021). Ground Robots for Inspection and Monitoring: A State-of-the-Art Review. *The IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. 1-4 Dec. New York, NY, USA. DOI: 10.1109/UEMCON53757.2021.9666648
16. Robotique agricole : repenser la mécanisation agricole. [Type of medium]. Available: <https://www.entraid.com/articles/robotique-agricole-opportunite-repenser-mecanisation-agricole>
17. U.S. Ground Forces Robotics and Autonomous Systems (RAS) and Artificial Intelligence (AI): Considerations for Congress. [Type of medium]. Available: <https://www.everycrsreport.com/reports/R45392.html>
18. I 10 Robot Più Strani Ed Utili Mai Costruiti. [Type of medium]. Available: <https://www.smartweek.it/10-robot-piu-strani-ed-utili-mai-costruiti/6/>
19. Boston Dynamics. [Type of medium]. Available: <https://www.bostondynamics.com/>
20. Mehndiratta, M., Kayacan, E. (2019). A constrained instantaneous learning approach for aerial package delivery robots: onboard implementation and experimental results. *Auton Robot* 43, P. 2209–2228. DOI:10.1007/s10514-019-09875-y
21. Hoang VuPhan, Hoon CheolPark. (2020). Mimicking nature’s flyers: a review of insect-inspired flying robots. *Current Opinion in Insect Science*. Volume 42, Pages 70-75. DOI: 10.1016/j.cois.2020.09.008.
22. Hao Zeng, Piotr Wasylczyk, Diederik S. Wiersma, Arri Priimagi. (2018). Light Robots: Bridging the Gap between Microrobotics and Photomechanics in Soft Materials. *Advanced Material. Special Issue: Materials Science in Finland*. Volume:30, Issue:24. DOI: 10.1002/adma.201703554

23. L. Angrisani; S. Grazioso; G. Di Gironimo; D. Panariello; A. Tedesco. (2019). On the use of soft continuum robots for remote measurement tasks in constrained environments: a brief overview of applications. *The IEEE International Symposium on Measurements & Networking (M&N)*. 8-10 July. Catania, Italy. DOI: 10.1109/IWMN.2019.8805050.
24. Farzeen Shahid, Jing-Shan Zhao, Pascal Godefroit. (2019). Design of flying robots inspired by the evolution of avian flight. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*. Volume: 233. Issue: 23-24, Page(s): 7669-7686. DOI: 10.1177/0954406219861995
25. Deterrence and defence. [Type of medium]. Available: https://www.nato.int/cps/en/natohq/topics_133127.htm
26. DJI - Official Website. [Type of medium]. Available: <https://www.dji.com/>
27. Mary B. Alatis; Gerhard P. Hancke.(2020). A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods. *IEEE Access*. Volume 8. Page(s): 39830 – 39846. DOI: 10.1109/ACCESS.2020.2975643.
28. Quentin Bramas, Pascal Lafourcade, Stéphane Devismes. (2020). Finding Water on Poleless Using Melomaniac Myopic Chameleon Robots. *Dagstuhl Research Online Publication Server*. pp.6:1–6:19. DOI: 10.5281/zenodo.3606387
29. Pop, Cornel Vlad; Tătar, Mihai Olimpiu. (2021). Studies and Research on the Locomotion of Mobile Robots Inspired By Biosystems. *Robotica & Management*. Vol. 26 Issue 2, P. 35-42.
30. Eugene Kagan, Nir Shvalb, Irad Ben-Gal. Autonomous Mobile Robots and Multi-Robot Systems. John&Son Ltd. 2020. P. 319.

31. lytec HQ2011 - 5 Intelligent Remote Control Nesting Boats. [Type of medium]. Available: https://www.gearbest.com/rc-boats/pp_009998728929.html
32. An Introduction to Underwater Drones . [Type of medium]. Available: <https://www.bhphotovideo.com/explora/video/tips-and-solutions/introduction-underwater-drones>
33. Janis Arents, Modris Greitans. (2022). Smart Industrial Robot Control Trends, Challenges and Opportunities within Manufacturing. *Appl. Sci.*, 12(2), 937. DOI: 10.3390/app12020937.
34. Inkulu, A.K., Bahubalendruni, M.V.A.R., Dara, A. and K., S. (2022). Challenges and opportunities in human robot collaboration context of Industry 4.0 - a state of the art review. *Industrial Robot*, Vol. 49 No. 2, pp. 226-239. DOI:10.1108/IR-04-2021-0077.
35. Ji, W., Wang, L. (2019) Industrial robotic machining: a review. *Int J Adv Manuf Technol* 103, 1239–1255. DOI:10.1007/s00170-019-03403-z.
36. KUKA linear robots. [Type of medium]. Available: <https://www.kuka.com/en-gb/products/robotics-systems/industrial-robots/linear-robots>
37. Industrial Desktop Style Scara Robot. [Type of medium]. Available: <https://www.motion-control.net/scara-robot/>
38. Промислові роботи. Класифікація промислових роботів. [Type of medium]. Available: https://kemppi.in.ua/ua/articles/promochlenie_roboti.htm
39. FANUC M-20ia- Robots Done Right. [Type of medium]. Available: <https://www.pinterest.com/pin/fanuc-m20ia-robots-done-right--614530311642893220/>
40. Zhi Feng, Guoqiang Hu, Yajuan Sun, Jeffrey Soon. (2020). An overview of collaborative robotic manipulation in multi-robot systems. *Annual Reviews in Control*. Volume 49. Pages 113-127. DOI: 10.1016/j.arcontrol.2020.02.002

41. Weiyong Si, Ning Wang, Chenguang Yang. (2020). A review on manipulation skill acquisition through teleoperation-based learning from demonstration. *Cognitive Computation and Systems*. Volume 3. Pages 1–16. DOI: 10.1049/ccs2.12005
42. Sinnemann, J., Boshoff, M., Dyrska, R. et al. (2022). Systematic literature review of applications and usage potentials for the combination of unmanned aerial vehicles and mobile robot manipulators in production systems. *Prod. Eng. Res. Devel.* DOI:10.1007/s11740-022-01109-y
43. Bodenhagen, L., Suvei, SD., Juel, W.K. et al. (2019). Robot technology for future welfare: meeting upcoming societal challenges – an outlook with offset in the development in Scandinavia. *Health Technol.* Volume 9, Pages 197–218 DOI:10.1007/s12553-019-00302-x
44. RoboSklep. Robotyka i Drony. Mobile Manipulators [Type of medium]. Available: <https://robosklep.com/en/41-mobile-manipulators>
45. Mobile manipulator FETCH [Type of medium]. Available: <https://robosklep.com/en/mobile-manipulators/119-mobile-manipulator-fetch-.html>
46. Mobile manipulator XL-GEN [Type of medium]. Available: <https://robosklep.com/en/mobile-manipulators/196-mobile-manipulator-xl-gen.html>
47. Mobile manipulator RB-VULCANO [Type of medium]. Available: <https://robosklep.com/en/mobile-manipulators/194-mobile-manipulator-rb-vulcano.html>
48. Mobile manipulator RB-EKEN [Type of medium]. Available: <https://robosklep.com/en/mobile-manipulators/197-mobile-manipulator-rb-eken.html>
49. Mobile manipulator RB-KAIROS [Type of medium]. Available: <https://robosklep.com/en/mobile-manipulators/204-mobile-manipulator-rb-vulcano.html>

50. Jaguar V6 with Manipulator Arm Mobile Robotic Platform [Type of medium]. Available: <https://robosklep.com/en/mobile-manipulators/173-jaguar-v6-with-manipulator-arm-mobile.html>

51. Dr. Robot Scout2 WiFi Mobile Development Platform (w/Dual arms). [Type of medium]. Available: <https://robosklep.com/en/mobile-manipulators/55-dr-robot-scout2.html>

52. Lucas Manuelli, Wei Gao, Peter Florence, Russ Tedrake (2019). Lucas Manuelli, Wei Gao, Peter Florence, Russ Tedrake. *Arxiv*. DOI:10.48550/arXiv.1903.06684

53. Fei Chen; Mario Selvaggio; Darwin G. Caldwell. (2019). Dexterous Grasping by Manipulability Selection for Mobile Manipulator With Visual Guidance. *IEEE Transactions on Industrial Informatics*. Volume: 15, Issue: 2. Page(s): 1202 – 1210. DOI: 10.1109/TII.2018.2879426

54. Riby Abraham Bobby, Alexandr Klimchik.(2021). Combination of geometric and parametric approaches for kinematic identification of an industrial robot. *Robotics and Computer-Integrated Manufacturing*. Volume 71. DOI: 10.1016/j.rcim.2021.102142

55. Yi Fang, Jin Qi, Jie Hu, WeimingWang, Yinghong Peng. (2020). An approach for jerk-continuous trajectory generation of robotic manipulators with kinematical constraints. *Mechanism and Machine Theory*. Volume 153. DOI: 10.1016/j.mechmachtheory.2020.103957

56. Jiankun Wang; Max Q.-H. Meng; Oussama Khatib. (2020). EB-RRT: Optimal Motion Planning for Mobile Robots. *IEEE Transactions on Automation Science and Engineering*. Volume: 17. Issue: 4. Page(s): 2063 – 2073. DOI: 10.1109/TASE.2020.2987397

57. Xuebo Zhang, Jiarui Wang, Yongchun Fang, Jing Yuan. (2019). Multilevel Humanlike Motion Planning for Mobile Robots in Complex Indoor Environments. *IEEE Transactions on Automation Science and Engineering*. Volume: 16, Issue: 3. Page(s): 1244 – 1258. DOI: 10.1109/TASE.2018.2880245.

58. Zheng Zhang, Xiangqi Ni, Weiliang Gao, Hongcheng Shen, Min Sun, Guodong Guo, Huaping Wu, and Shaofei Jiang.(2021). Pneumatically Controlled Reconfigurable Bistable Bionic Flower for Robotic Gripper. *Soft Robotics*. DOI: 10.1089/soro.2020.0200

59. Sohyun Kim; Youngsu Cha. (2021). Double-layered electrohydraulic actuator for bi-directional bending motion of soft gripper. *The 18th International Conference on Ubiquitous Robots (UR)*. 12-14 July. Gangneung, Korea (South). DOI: 10.1109/UR52253.2021.9494690.

60. Dan Dumitriu, Marius Ionescu, Daniel Octavian Melinte, Mihai Mărgăritescu. Stroke Enhancement Adapter for Electric Parallel Gripper Equipping a Mobile X-Y-Z Robotic System. *Transilvania university press of braşov*. P. 49-52

61. Tristan Abondance; Kaushik Jayaram; Noah T. Jafferis; Jennifer Shum; Robert J. Wood. (2020). Piezoelectric Grippers for Mobile Micromanipulation. *IEEE Robotics and Automation Letters*. Volume: 5, Issue: 3. Page(s): 4407 – 4414. DOI: 10.1109/LRA.2020.2997317

62. Şerdean, M., Şerdean, F., Mândru, D. (2021). An Overview of Grippers in Agriculture Robotic Systems. In: Lovasz, EC., Maniu, I., Doroftei, I., Ivanescu, M., Gruescu, CM. (eds) *New Advances in Mechanisms, Mechanical Transmissions and Robotics . MTM&Robotics 2020. Mechanisms and Machine Science*, vol 88. Springer, Cham. DOI:10.1007/978-3-030-60076-1_19

63. Can Özbaran; Savaş Dilibal; Gökem Sungur. (2020). Mechatronic System Design of A Smart Mobile Warehouse Robot for Automated Storage/Retrieval Systems. *The Innovations in Intelligent Systems and Applications Conference (ASYU)*. 15-17 Oct. Istanbul, Turkey. DOI: 10.1109/ASYU50717.2020.9259882

64. Ionel-Alexandru Gal, Alexandra-Cătălina Ciocîrlan and Mihai Mărgăritescu. (2021). State Machine-Based Hybrid Position/Force Control Architecture for a Waste Management Mobile Robot with 5DOF Manipulator. *Applied Sciences*, Vol 11, No. 9. DOI: 10.3390/app11094222

65. Kereshmeh Afsari, Sahil Gupta, Mahdi Afkhamiaghda and Ziyang Lu. (2018). Applications of Collaborative Industrial Robots in Building Construction. *The 54th ASC Annual International Conference Proceedings*. P.472-479.
66. Vu HuyNguyen, Won-JongKim. (2020). Error analysis to minimize cross-axis couplings in 6-DOF motion systems with a single moving part. *Precision Engineering*. Volume 63. Pages 49-61. DOI: 10.1016/j.precisioneng.2019.11.013.
67. Marco Faroni; Manuel Beschi; Nicola Pedrocchi (2020). Inverse Kinematics of Redundant Manipulators With Dynamic Bounds on Joint Movements. *IEEE Robotics and Automation Letters*, Volume: 5, Issue: 4. Page(s): 6435 – 6442. DOI: 10.1109/LRA.2020.3013879
68. Chaoyang Tan; Xiaoqin Gu; Xianfeng Zhang. (2019). A Dynamics Control Technology of the Redundant Space Manipulator. *The International Conference on Advanced Mechatronic Systems (ICAMechS)*. 26-28 Aug. Kusatsu, Japan. DOI: 10.1109/ICAMechS.2019.8861639
69. Quoc-Cuong Nguyen, Viet-Hung Vu, Marc Thomas. (2022). A Kalman filter based ARX time series modeling for force identification on flexible manipulators. *Mechanical Systems and Signal Processing*. Volume 169. DOI: 10.1016/j.ymssp.2021.108743.
70. Jonathan Obregón-Flores; Gustavo Arechavaleta; Héctor M. Becerra; América Morales-Díaz. (2021). Predefined-Time Robust Hierarchical Inverse Dynamics on Torque-Controlled Redundant Manipulators. *IEEE Transactions on Robotics*, Volume: 37, Issue: 3. Page(s): 962 – 978. DOI: 10.1109/TRO.2020.3042054
71. Maarroof, Omar W., Mehmet İ.C. Dede, and Levent Aydin. (2022). A Robot Arm Design Optimization Method by Using a Kinematic Redundancy Resolution Technique. *Robotics*. Volume 11, No. 1. DOI:10.3390/robotics11010001.

72. Dequan Zhang, Xu Han. (2020). Kinematic Reliability Analysis of Robotic Manipulator. *Journal of Mechanical Design*. 142(4). DOI: 10.1115/1.4044436.
73. Klug, C., Schmalstieg, D., Gloor, T. et al. (2019). A Complete Workflow for Automatic Forward Kinematics Model Extraction of Robotic Total Stations Using the Denavit-Hartenberg Convention. *J Intell Robot Syst*. 95, P.311–329. DOI:10.1007/s10846-018-0931-4
74. Carlos Faria; João L. Vilaça; Sérgio Monteiro; Wolfram Erlhagen; Estela Bicho. (2019). Automatic Denavit-Hartenberg Parameter Identification for Serial Manipulators. *The 45th Annual Conference of the IEEE Industrial Electronics Society*. 14-17 Oct. Lisbon, Portugal. DOI: 10.1109/IECON.2019.8927455
75. Guanglei Wu, Huiping Shen. Parallel PnP Robots. Parametric Modeling, Performance Evaluation and Design Optimization. *Springer*. P.262. ISSN 2523-3386. DOI: 10.1007/978-981-15-6671-4
76. Martín-Martín, Roberto. Leveraging Problem Structure in Interactive Perception for Robot Manipulation of Constrained Mechanisms. *Technische Universitaet Berlin (Germany), ProQuest Dissertations Publishing*, 2018. P.24. DOI: 10.14279/depositonce-6732.
77. Binbin Lian, Xi Vincent, Wang, Lihui Wang. (2019). Static and dynamic optimization of a pose adjusting mechanism considering parameter changes during construction. *Robotics and Computer-Integrated Manufacturing*, Volume 59, Pages 267-277. DOI: 10.1016/j.rcim.2019.04.008.
78. Edward J. Haug. (2022). Manipulator Kinematics and Dynamics on Differentiable Manifolds: Part II Dynamics. *J. Comput. Nonlinear Dynam*, Volume 17(2). DOI: 10.1115/1.4052653
79. Jing, Z., Xu, Q. & Huang, J. (2019). A review on kinematic analysis and dynamic stable control of space flexible manipulators. *AS*. Volume 2, Pages 1–14 DOI:10.1007/s42401-018-00024-4.

80. Cao, F., Docherty, P.D. & Chen, X. (2022). Contact force estimation for serial manipulator based on weighted moving average with variable span and standard Kalman filter with automatic tuning. *Int J Adv Manuf Technol*. Volume 118, Pages 3443–3456. DOI:10.1007/s00170-021-08036-9.
81. Abo-Shanab, R.F. (2020). Dynamic modeling of parallel manipulators based on Lagrange–D’Alembert formulation and Jacobian/Hessian matrices. *Multibody Syst Dyn*, Volume 48, Pages 403–426. DOI:10.1007/s11044-019-09705-0
82. Kheylo, S.V., Tsarkov, A.V., Garin, O.A. (2020). Kinematic Analysis of Novel 6-DOF Robot. In: Hu, Z., Petoukhov, S., He, M. (eds) *Advances in Artificial Systems for Medicine and Education III. AIMEE 2019. Advances in Intelligent Systems and Computing*, vol 1126. Springer, Cham. DOI:10.1007/978-3-030-39162-1_40
83. Sholanov, K.S. (2021). Parallel Manipulators’ Dynamics. In: *Parallel Manipulators of Robots. Mechanisms and Machine Science*, Vol 92, P. 119–132. DOI:10.1007/978-3-030-56073-7_7
84. Maria Vittoria Minniti; Farbod Farshidian; Ruben Grandia; Marco Hutter. (2019). Whole-Body MPC for a Dynamically Stable Mobile Manipulator. *IEEE Robotics and Automation Letters*. Volume: 4, Issue: 4, Page(s): 3687 – 3694. DOI: 10.1109/LRA.2019.2927955.
85. Cezara Coman, Mario Ivan, Cozmin Cristoiu, Bogdan Verdete. (2018). Dedicated Forward Kinematics Algorithms for SCARA and Palletizing Robots. *Proceedings in Manufacturing Systems*, Volume 13, Issue 4, P.189-194
86. Stefan Segla. (2018). Static Balancing of Robot Mechanisms and Manipulation Devices. *Journal of Mechanical Engineering – Strojnícky časopis*, Vol 68, No 2, P.77 – 90. DOI: 10.2478/scjme-2018-0019.
87. Vezvari, M.R., Nikoobin, A. & Ghoddosian, A. (2020) Zero-power balancing a two-link robot manipulator for a predefined point-to-point task. *J Mech Sci Technol*. Vol. 34, P. 2585–2595. DOI:10.1007/s12206-020-0533-5

88. Qu, X., Liu, Q., Wang, H. et al. (2022). A spiral path generation method for achieving uniform material removal depth in aspheric surface polishing. *Int J Adv Manuf Technol*. Vol 119, P.3247–3263. DOI:10.1007/s00170-021-08420-5
89. Ameer Hamza Khan; Shuai Li; Xin Luo. (2020). Obstacle Avoidance and Tracking Control of Redundant Robotic Manipulator: An RNN-Based Metaheuristic Approach. *IEEE Transactions on Industrial Informatics*. Volume: 16, Issue: 7, Page(s): 4670 – 4680. DOI: 10.1109/TII.2019.2941916.
90. Changyeob Shin; Peter Walker Ferguson; Sahba Aghajani Pedram; Ji Ma; Erik P. Dutson; Jacob Rosen.(2019). Autonomous Tissue Manipulation via Surgical Robot Using Learning Based Model Predictive Control. *The International Conference on Robotics and Automation (ICRA)*. 20-24 May. Montreal, QC, Canada. DOI: 10.1109/ICRA.2019.8794159.
91. Vorob'eva, N.S., Zhoga, V.V. & Nesmiyanov, I.A. (2019). Program Displacement Tracing of Executive Devices by the Manipulator Drives of Parallel-Sequential Structures. *J. Comput. Syst. Sci. Int.* 58, P. 305–316. DOI:10.1134/S1064230719020187
92. Hichri, B., Adouane, L., Fauroux, J.-C., Mezouar, Y. and Doroftei, I. (2019). Flexible co-manipulation and transportation with mobile multi-robot system. *Assembly Automation*, Vol. 39 No. 3, pp. 422-431. DOI:10.1108/AA-12-2017-190.
93. Zhenhui Zhan, Xianmin Zhang, Haodong Zhang, Gengchao Chen. (2019). Unified motion reliability analysis and comparison study of planar parallel manipulators with interval joint clearance variables. *Mechanism and Machine Theory*. Volume 138, Pages 58-75. DOI: 10.1016/j.mechmachtheory.2019.03.041.
94. Zonggao Mu; Han Yuan; Wenfu Xu; Tianliang Liu; Bin Liang. (2020). A Segmented Geometry Method for Kinematics and Configuration Planning of Spatial Hyper-Redundant Manipulators. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Volume: 50, Issue: 5, Page(s): 1746 – 1756. DOI: 10.1109/TSMC.2017.2784828

95. Fan Xiao, Gongfa Liab, Du Jianga, Yuanmin Xie, Juntong Yun, Ying Liub, Li Huang, Zifan Fang. (2021). An effective and unified method to derive the inverse kinematics formulas of general six-DOF manipulator with simple geometry. *Mechanism and Machine Theory*. Volume 159. DOI: 10.1016/j.mechmachtheory.2021.104265.
96. Zhan Zhenhui, Zhang Xianmin, Jian Zhicong, Zhang Haodong. (2018). Error modelling and motion reliability analysis of a planar parallel manipulator with multiple uncertainties. *Mechanism and Machine Theory*. Volume 124, Pages 55-72. DOI: 10.1016/j.mechmachtheory.2018.02.005.
97. Lang Yan, Xinghua Chang, Nianhua Wang, Runyu Tian, Laiping Zhang, Wei Liu.(2021). Learning how to avoid obstacles: A numerical investigation for maneuvering of self-propelled fish based on deep reinforcement learning. *Internation Journal for Numerical Methods in Fluids*. Volume 93, Issue 10, Pages 3073-3091. DOI: 10.1002/flid.5025
98. Chen Li; Hui Dong; Xubing Li; Weikang Zhang; Xiaodong Liu; Ligang Yao; Hao Sun. (2021). Inverse Kinematics Study for Intelligent Agriculture Robot Development via Differential Evolution Algorithm. *The International Conference on Computer, Control and Robotics (ICCCR)*. 8-10 Jan. Shanghai, China. DOI: 10.1109/ICCCR49711.2021.9349277.
99. Liu, P., Huda, M.N., Tang, Z. et al. (2020). A self-propelled robotic system with a visco-elastic joint: dynamics and motion analysis. *Engineering with Computers*. Volume 36, Pages 655–669. DOI:10.1007/s00366-019-00722-3
100. Wenfu Xu; Tianliang Liu; Yangmin Li. (2018). Kinematics, Dynamics, and Control of a Cable-Driven Hyper-Redundant Manipulator. *IEEE/ASME Transactions on Mechatronics*. Volume: 23, Issue: 4, Page(s): 1693 – 1704. DOI: 10.1109/TMECH.2018.2842141
101. Nina R. Sinatra, Clark B. Teeple, Daniel M. Vogt, Kevin Kit Parker, David F. Gruber, Robert J. Wood. (2019). Ultragentle manipulation of delicate structures using a soft robotic gripper. *Science Robotics*. Volume 4, Issue 33. DOI:10.1126/scirobotics.aax5425

102. Christos K. Verginis; Matteo Mastellaro; Dimos V. Dimarogonas. (2020). Robust Cooperative Manipulation Without Force/Torque Measurements: Control Design and Experiments. *IEEE Transactions on Control Systems Technology*. Volume 28, Issue 3, Page(s): 713 – 729. DOI: 10.1109/TCST.2018.2885682
103. Bing Xiao; Shen Yin. (2019). Exponential Tracking Control of Robotic Manipulators With Uncertain Dynamics and Kinematics. *IEEE Transactions on Industrial Informatics*. Volume 15, Issue 2. Page(s): 689 – 698. DOI: 10.1109/TII.2018.2809514
104. Andrej Cibicik, Eilif Pedersen, Olav Egeland. (2020). Dynamics of luffing motion of a flexible knuckle boom crane actuated by hydraulic cylinders. *Mechanism and Machine Theory*. Volume 143. DOI: 10.1016/j.mechmachtheory.2019.103616.
105. Zhang, HQ., Fang, HR., Jiang, BS. et al. (2019). Dynamic Performance Evaluation of a Redundantly Actuated and Over-constrained Parallel Manipulator. *Int. J. Autom. Comput.* Volume 16, P. 274–285. DOI:10.1007/s11633-018-1147-6
106. Wenjuan He.(2021). WITHDRAWN: All-Round Development of Mooc English Teachers Based on Sensor Path Coverage Algorithm and Microprocessor. *Microprocessors and Microsystems*. DOI: 10.1016/j.micpro.2021.104068
107. Princewill Akpojotor, Adebayo Adetunmbi, Boniface Alese, Ayodeji Oluwatope. (2020). Automatic license plate recognition on microprocessors and custom computing platforms: A review. *IET Image Processing*. P. 1–19. DOI: 10.1049/ipr2.12262.
108. G. Glenn Henry. (2021). From Mainframes to Microprocessors. *IEEE Micro*. Volume: 41, Issue: 6. Page(s): 89 – 96. DOI: 10.1109/MM.2021.3112877

109. Stephen Mallo. (2022). The Challenges of Lecture Delivery of Arm X86, Cisc and Risc in the Teaching of Coursecsc303 (Computer Architecture) in the University of Jos, Nigeria: an Overview. *International Journal of Advanced Engineering and Management Research*. Vol. 7, No. 02. P.148-159. ISSN: 2456-3676.
110. Steven Van Singel. (2021). The Renesas Automotive Story in the History of the Microprocessor. *IEEE Micro*. Volume: 41, Issue: 6, Page(s): 107 – 108. DOI: 10.1109/MM.2021.3113821
111. Malti Bansall and Harsh. (2021). Reduced Instruction Set Computer (RISC): A Survey. *Journal of Physics: Conference Series*. Volume 1916, P.1-14. DOI: 10.1088/1742-6596/1916/1/012280.
112. Nauman Riaz Chaudhry, Anastasia Anagnostou, Simon J. E. Taylor. (2022). A Workflow Architecture for Cloud-based Distributed Simulation. *ACM Transactions on Modeling and Computer Simulation*, Volume 32, Issue 2, Article No.: 15, Pp: 1–26. DOI:10.1145/3503510.
113. Hanh, T.T.H., Doucet, A., Sidere, N., Moreno, J.G., Pollak, S. (2021). Named Entity Recognition Architecture Combining Contextual and Global Features. In: Ke, HR., Lee, C.S., Sugiyama, K. (eds) Towards Open and Trustworthy Digital Societies. ICADL 2021. Lecture Notes in Computer Science(), vol 13133. Springer, Cham. DOI:10.1007/978-3-030-91669-5_21.
114. Lee, Jongbok. (2021). VHDL Design for Out-of-Order Superscalar Processor of A Fully Pipelined Scheme. *The Journal of the Institute of Internet, Broadcasting and Communication*. Volume 21, Issue 1, Pages.99-105.
115. Malik Imran, Felipe Almeida, Jaan Raik, Andrea Basso, Sujoy Sinha Roy, Samuel Pagliarini. (2021). Design Space Exploration of SABER in 65nm ASIC. *ASHES '21: Proceedings of the 5th Workshop on Attacks and Solutions in Hardware Security*, Pages 85–90. DOI: 10.1145/3474376.3487278

116. Weiqi Zhi; Ting Yi; Zhiliang Hong. (2021). A Review and Perspective on Electrode Patch-Based Fetal ECG Monitoring ASIC. *The IEEE 14th International Conference on ASIC (ASICON)*. 26-29 Oct. Kunming, China. DOI: 10.1109/ASICON52560.2021.9620269
117. Kilinceker, O., Turk, E., Belli, F. et al. (2021). Model-based ideal testing of hardware description language (HDL) programs. *Softw Syst Model*. DOI:10.1007/s10270-021-00934-6.
118. N. T. Gadawe, T.A. Fathi, S.L. Qaddoori and R. W. Hamad. (2021). Synthesis and Implementation of IIR Filter using VHDL Language. *IOP Conference Series: Materials Science and Engineering*, Volume 1152
119. Kato, M., Onizawa, N., & Hanyu, T. (2021). Design automation of invertible logic circuit from a standard hdl description. *If CoLoG Journal of Logics and their Applications*, 8(5), Pp.1311-1333.
120. Muhammad Yusro, Nanang Arif Guntoro, and Rikawarastuti. (2021). Utilization of microcontroller technology using Arduino board for Internet of Things (a systematic review). *AIP Conference Proceedings*, Volume 2331, Issue 1, DOI: 10.1063/5.0041705.
121. Mijailović, Đorđe, Aleksandar Đorđević, Miladin Stefanovic, Dejan Vidojević, Albina Gazizulina, and Damir Projović. (2021). A Cloud-Based with Microcontroller Platforms System Designed to Educate Students within Digitalization and the Industry 4.0 Paradigm. *Sustainability*. Volume 13, No. 22: DOI: 10.3390/su132212396.
122. Yang, P., Lai, S., Guan, H. & Wang, J. (2022). Teaching Reform and Practice Using the Concept of Outcome-Based Education: A Case Study on Curriculum Design for a Microcontroller Unit Course. *International Journal of Emerging Technologies in Learning (iJET)*, 17(3), 68-82.

123. Diego Fernando Ramirez Jimenez; Alexander Lopez Parrado; Jaime Velasco Medina. (2021). Overview of a framework for Implementation of digital controllers in Energia IDE using Texas Instruments microcontrollers. *The IEEE 5th Colombian Conference on Automatic Control (CCAC)*. 19-22 Oct. Ibagué, Colombia. DOI: 10.1109/CCAC51819.2021.9633305.
124. Hilario A. Calinao Jr., MS-ECE. (2021). Development of Auto-Generated Code for Programming PIC16F877A Microcontroller for Microprocessor System Subject. *IECEP Journal*. Volume 4, No. 1. P.38-49.
125. Hur, B. (2021). Self-publish textbook for Embedded System Education using an MSP432 microcontroller. *The Paper presented at ASEE 2021 Gulf-Southwest Annual Conference*, Waco, Texas. <https://peer.asee.org/36399>.
126. Erfan Abbasian, Morteza Gholipour, Farzaneh Izadinasab. (2021). Performance evaluation of GNRfet and TMDfet devices in static random access memory cells design. *International Journal of Circuit Theory and Applications*. Volume49, Issue11. DOI: /10.1002/cta.3108.
127. Chia-Lung Hung, Bing-Yue Tsui, Te-Kai Tsai, Li-Jung Lin and Yu-Xin Wen. (2022). Design, Process, and Characterization of Complementary Metal–Oxide–Semiconductor Circuits and Six-Transistor Static Random-Access Memory in 4H-SiC. *ECS Journal of Solid State Science and Technology*, Volume 11, Number 4, P.345-367. DOI: 10.1149/2162-8777/ac6119.
128. Deepam Goyal; Chirag Mongia; Shankar Sehgal. (2021). Applications of Digital Signal Processing in Monitoring Machining Processes and Rotary Components: A Review. *IEEE Sensors Journal*, Volume: 21, Issue: 7. Page(s): 8780 – 8804. DOI: 10.1109/JSEN.2021.3050718.
129. Ales Prochazka; Oldrich Vysata; Vladimir Marik. (2021). Integrating the Role of Computational Intelligence and Digital Signal Processing in Education: Emerging Technologies and Mathematical Tools. *IEEE Signal Processing Magazine*, Volume: 38, Issue: 3, Page(s): 154 – 162. DOI: 10.1109/MSP.2021.3058634.

130. Ezio Bartocci, Luca Bortolussi, Michele Loreti, Laura Nenzi, and Simone Silveti. (2020). MoonLight: A Lightweight Tool for Monitoring Spatio-Temporal Properties. In Runtime Verification, Jyotirmoy Deshmukh and Dejan Ničković (Eds.). *Springer International Publishing*, Cham, 417-428.
131. Jiahui Cao; Shaohua Tian; Zhibo Yang; Guangrong Teng; Haoqi Li; Ruochen Jin; Ruqiang Yan; Xuefeng Chen. (2022). Blade Tip Timing Signal Filtering Method Based on Sampling Aliasing Frequency Map. *IEEE Transactions on Instrumentation and Measurement*. Volume: 71. DOI: 10.1109/TIM.2022.3156979.
132. Ling Xu, Feiyan Chen, Feng Ding, Ahmed Alsaedi, Tasawar Hayat. (2021). Hierarchical recursive signal modeling for multifrequency signals based on discrete measured data. *International Journal of Adaptive Control and Signal Processing*. Volume 35, Issue 5. Pages 676-693. DOI: 10.1002/acs.3221
133. Roghayyeh Arvanaghi, Sebelan Danishvar, Morad Danishvar. (2022). Classification cardiac beats using arterial blood pressure signal based on discrete wavelet transform and deep convolutional neural network. *Biomedical Signal Processing and Control*. Volume 71, Part A. DOI: 10.1016/j.bspc.2021.103131.
134. Sanjeevi, M; Jayanthi, D L. (2021). Transportation Emission Level Prediction Using Mobile. *i-Manager's Journal on Electronics Engineering; Nagercoil*. Vol. 11, Iss. 3, P: 6-11. DOI: 10.26634/jele.11.3.18459.
135. Yan Bai, Jie Xi, Wendong Liao. (2021). Design and implementation of full-body motion capture system based on multi-sensor fusion. *The International Conference on Neural Networks, Information and Communication Engineering*, 2021, Qingdao, China. DOI: 10.1117/12.2615157.
136. Jianfeng Yao; Yiwei Liu; Handong Hu; Kening Gong; Yang Liu. (2021). Research on Double-Motor Cooperative Control of Dexterous Finger with Variable Stiffness. *The 4th International Conference on Mechatronics, Robotics and Automation (ICMRA)*. 22-24 Oct. Zhanjiang, China. DOI: 10.1109/ICMRA53481.2021.9675699.

137. Haibo Wei; Jianwei Mei; Jie Liu; Zhaozhao Wang; Shengli Fang; Long Wang. (2021). Application Research on Data Reliability of SPI Bus between Microprocessors. *The IEEE 4th International Conference on Electronics Technology (ICET)*. 7-10 May. Chengdu, China. DOI: 10.1109/ICET51757.2021.9450983.
138. Meroth, A., Sora, P. (2021). Serial Peripheral Interface (SPI). In: *Sensornetzwerke in Theorie und Praxis*. Springer Vieweg, Wiesbaden. DOI: 10.1007/978-3-658-31709-6_8.
139. Bitty Jose; J. Samson Immanuel. (2021). Design of BIST(Built-In-Self-Test) Embedded Master-Slave communication using SPI Protocol. *The 3rd International Conference on Signal Processing and Communication (ICPSC)*. 13-14 May. Coimbatore, India. DOI: 10.1109/ICSPC51351.2021.9451702.
140. Gazi, O., Arlı, A.Ç. (2021). Inter Integrated Circuit (I2C) Serial Communication in VHDL. In: *State Machines using VHDL*. Springer, Cham. https://doi.org/10.1007/978-3-030-61698-4_5
141. Mark Eduard Gross, Dorin Marius Petreus. (2021). Data Monitoring And Acquisition System For The I²C Protocol. *ACTA Technica Napocensis Electronics and Telecommunications*. Volume 61, P.7- 10
142. Texas Instruments – Understanding the I2C Bus, [Type of medium]. Available: https://www.ti.com/lit/an/slva704/slva704.pdf?ts=1615100127956&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FTCA9548A
143. Salman Ahmad; Atif Iqbal; Mohammad Ali; Khaliquir Rahman; Abdellahi Sidi Ahmed. (2021). A Fast Convergent Homotopy Perturbation Method for Solving Selective Harmonics Elimination PWM Problem in Multi Level Inverter. *IEEE Access*. Volume 9. Page(s): 113040 – 113051. DOI: 10.1109/ACCESS.2021.3104184.

144. Shuvra Prokash Biswas; Md. Shamim Anower; Md. Rafiqul Islam Sheikh; Md. Rabiul Islam; Md. Ashib Rahman; M A Parvez Mahmud. (2021). A Modified Reference Saturated Third Harmonic Injected Equal Loading PWM for VSC-Based Renewable Energy Systems. *IEEE Transactions on Applied Superconductivity*, Volume: 31, Issue: 8, Article Sequence Number: 5000405. DOI: 10.1109/TASC.2021.3096484.

145. Marius Takongmo; Chenhui Zhang; John Salmon. (2021). Parallel Inverters using a DC Common Mode PWM Filter with an AC Differential Mode PWM Filter. *The IEEE Applied Power Electronics Conference and Exposition (APEC)*. 14-17 June. Phoenix, AZ, USA. DOI: 10.1109/APEC42165.2021.9487453.

146. Tinker Board, [Type of medium]. Available: <https://www.asus.com/uk/Networking-IoT-Servers/AIoT-Industrial-Solutions/All-series/Tinker-Board/>

147. ASUS SBC Tinker board RK3288 SoC. [Type of medium]. Available: <https://www.amazon.com/ASUS-Tinker-board-RK3288-Mali-T764/dp/B06VSBVQWS>

148. Tinker Board 2. [Type of medium]. Available: <https://tinker-board.asus.com/product/tinker-board-2.html>.

149. Raspberry Pi 3 Model A+. [Type of medium]. Available: <https://www.raspberrypi.com/products/raspberry-pi-3-model-a-plus/>.

150. Raspberry PI 3 model A+ - Raspberry Pi 3 - Model A+ 512MB RAM. [Type of medium]. Available: <https://www.distrelec.biz/en/raspberry-pi-model-512mb-ram-raspberry-pi-raspberry-pi-model/p/30127220>.

151. G.Samrat Krishna, T.Srinivasa Ravi Kiran, A.Srisaila. (2021). Testing performance of RaspberryPi as IDS using SNORT. *Materialstoday Proceedings*. DOI: 10.1016/j.matpr.2021.01.607

152. Raspberry Pi 3 Model B+. The final revision of our third-generation single-board computer. [Type of medium]. Available: <https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/>

153. 26 Awesome Uses for a Raspberry Pi. Which Raspberry Pi project should you start with? Here's our roundup of the best Raspberry Pi uses and projects around! [Type of medium]. Available: <https://www.makeuseof.com/tag/different-uses-raspberry-pi/>.

154. Raspberry Pi type A, 256 MB. [Type of medium]. Available: <https://www.elfa.se/en/raspberry-pi-type-256-mb-raspberry-pi-raspberry-pi-type/p/11044252>

155. Raspberry Pi Model A (256MB). [Type of medium]. Available: https://www.amazon.com/Raspberry-Pi-Model-A-256MB/product-reviews/B00BC0ZL88?reviewerType=all_reviews.

156. Raspberry Pi Zero. [Type of medium]. Available: <https://www.raspberrypi.com/products/raspberry-pi-zero/>.

157. Waveshare Raspberry Pi Zero V1.3 Low-Cost Pared Down Pi Half Size of a Model A+ BCM2835. [Type of medium]. Available: <https://www.amazon.com/Waveshare-Raspberry-Low-Cost-BCM2835-Affordable/dp/B01GEHPI0E>

158. Raspberry Pi Zero W. [Type of medium]. Available: <https://www.raspberrypi.com/products/raspberry-pi-zero-w/>

159. Raspberry Pi Zero WH v.1.1. [Type of medium]. Available: <https://www.distrelec.biz/en/raspberry-pi-zero-wh-raspberry-pi-raspberry-pi-zero-wh/p/30113445>

160. Orange Pi. [Type of medium]. Available: http://www.orangepi.org/OrangePiWin_WinPlus/

161. Orange Pi Win Plus A64 Quad-core 2GB DDR3 Development Board WiFi + Bluetooth. [Type of medium]. Available: <https://picclick.co.uk/Orange-Pi-Win-Plus-A64-Quad-core-2GB-DDR3-143013047326.html>

162. What's Orange Pi Zero. [Type of medium]. Available: <http://www.orangepi.org/orangepizero/>

163. Orange Pi Zero 256 MB (H3). [Type of medium]. Available: <https://www.robotistan.com/orange-pi-zero-256-mb-h3>

164. Orange Pi Pc Plus AllWinner H3 Run Android Ubuntu Debian (PC Plus). [Type of medium]. Available: <https://www.amazon.com/Orange-Pi-Raspberry-Wholesale-Available/dp/B07ZYLCC6T>
165. Orange Pi PC Plus. [Type of medium]. Available: <https://www.robotistan.com/orange-pi-pc-plus-board>.
166. LattePanda 2G/32G. [Type of medium]. Available: <https://www.lattepanda.com/products/1.html>.
167. LattePanda 2G/32GB With Windows 10 Activated. [Type of medium]. Available: <https://www.lattepanda.com/products/2.html>.
168. ODROID XU4Q with Passive Heatsink and Power Supply. [Type of medium]. Available: <https://www.amazon.com/ODROID-Passive-Heatsink-Power-Supply/dp/B0761YN732>
169. ODROID Motherboards. [Type of medium]. Available: <https://en.odroid.se/>
170. ODROID-XU4 Special Price. [Type of medium]. Available: <https://www.hardkernel.com/shop/odroid-xu4-special-price/>
171. ODROID-XU4 Single Board Computer with Quad Core. [Type of medium]. Available: <https://www.amazon.com/ODROID-XU4-Single-Board-Computer-Gigabit/dp/B0163GEA64>
172. ODROID HC2 : Home Cloud Two. [Type of medium]. Available: <https://www.amazon.com/ODROID-HC2-Home-Cloud-Two/dp/B0794DG2WF>
173. ODROID-HC2 as an entry-level NAS. [Type of medium]. Available: <https://loganmarchione.com/2018/06/odroid-hc2-as-an-entry-level-nas/>
174. PINE A64+. [Type of medium]. Available: <https://www.pine64.org/devices/single-board-computers/pine-a64/>
175. Pine64 PINE A64+ 1GB Board. [Type of medium]. Available: <https://www.fabtolab.com/pine64-a64-plus-1gb-board>.
176. PINE A64-LTS. [Type of medium]. Available: <https://www.pine64.org/devices/single-board-computers/pine-a64-lts/>

177. PINE A64-LTS Single Board Computer. [Type of medium]. Available: <https://pine64.com/product/pine-a64-lts/>.
178. ESP32-CAM Video Streaming and Face Recognition with Arduino IDE. [Type of medium]. Available: <https://randomnerdtutorials.com/esp32-cam-video-streaming-face-recognition-arduino-ide/>
179. ESP32-CAM: Machine Vision Tips, Camera Guides and Projects. [Type of medium]. Available: <https://www.arducam.com/esp32-machine-vision-learning-guide/>
180. ESP32-CAM, Camera Module Based on ESP32. [Type of medium]. Available: <https://www.waveshare.com/esp32-cam.htm>
181. Program and upload code to ESP32-CAM using FT232RL Adapter. [Type of medium]. Available: <https://www.elec-cafe.com/program-and-upload-code-to-esp32-cam-using-ft232rl-adapter/>
182. Arduino IDE 1.8.19. [Type of medium]. Available: <https://www.arduino.cc/en/software/>.
183. Gravity: Arduino Shield for Raspberry Pi B+/2B/3B/3B+/4B. [Type of medium]. Available: <https://www.dfrobot.com/product-1211.html>.
184. DFROBOT Shield for Raspberry Pi B+/2B/3B for Arduino. [Type of medium]. Available: <https://www.amazon.com/DFROBOT-Shield-Based-Arduino-Raspberry/dp/B01IKGUORE>.
185. Gravity: MCP23017 I2C 16 Digital IO Expansion Module. [Type of medium]. Available: <https://www.dfrobot.com/product-2002.html>
186. Waveshare MCP23017 IO Expansion Board. [Type of medium]. Available: <https://www.youtube.com/watch?v=YA6L9VdyJSw>.
187. POE_BOARD. [Type of medium]. Available: <https://www.digikey.gr/en/products/detail/raspberry-pi/POE-BOARD/8641689>
188. Li-ion Battery HAT for Raspberry Pi, 5V Output, Quick Charge. [Type of medium]. Available: <https://www.waveshare.com/li-ion-battery-hat.htm>

189. DVK512, Raspberry Pi Expansion Board. [Type of medium]. Available: <https://store.open-electronics.org/dvk512-raspberry-pi-expansion-board-2846-dvk512-futura-group-srl>.
190. 3 Channel Relay Module Shield Smart Home for Raspberry Pi. [Type of medium]. Available: <https://core-electronics.com.au/3-channel-relay-module-shield-smart-home-for-raspberry-pi-3.html>.
191. MC33886 Raspberry Pi Motor Driver Board for Raspberry Pi. [Type of medium]. Available: <https://www.robotshop.com/en/mc33886-raspberry-pi-motor-driver-board-raspberry-pi.html>.
192. RPi Motor Driver Board. [Type of medium]. Available: <https://www.waveshare.com/rpi-motor-driver-board.htm>
193. Raspberry Pi High-Precision ADC/DAC Board. [Type of medium]. Available: <https://www.digikey.com/catalog/en/partgroup/raspberry-pi-high-precision-adc-dac-board/65958>.
194. Raspberry Pi High-Precision AD/DA Expansion Board. [Type of medium]. Available: <https://www.waveshare.com/high-precision-ad-da-board.htm>.
195. GSM/GPRS/GNSS/Bluetooth HAT for Raspberry Pi. [Type of medium]. Available: <https://www.waveshare.com/gsm-gprs-gnss-hat.htm>.
196. Adding a Real Time Clock to Raspberry Pi. [Type of medium]. Available: <https://learn.adafruit.com/adding-a-real-time-clock-to-raspberry-pi>
197. Adding a Real Time Clock to your Raspberry Pi. [Type of medium]. Available: <https://thepihut.com/blogs/raspberry-pi-tutorials/17209332-adding-a-real-time-clock-to-your-raspberry-pi>
198. GY-291 ADXL345 3-AXIS Accelerometer. [Type of medium]. Available: <https://www.robotpark.com/GY-291-ADXL345-3-Axis-Accelerometer>.
199. ADXL345 3-axis Accelerometer (GY-291). [Type of medium]. Available: <https://opencircuit.shop/product/adxl345-3-axis-accelerometer-gy-291>.
200. Sharp GP2Y0A21YK0F IR Range Sensor - 10cm to 80cm. [Type of medium]. Available: <https://www.robotshop.com/uk/sharp-gp2y0a21yk0f-ir-range-sensor.html>.

201. IR Distance Sensor GP2Y0A21YK0F. [Type of medium]. Available: <https://www.smart-prototyping.com/Sharp-IR-Distance-Sensor-GP2Y0A21YK0F-10-80-cm>.
202. Sharp GP2Y0A41SK0F Analog Distance Sensor 4-30cm. [Type of medium]. Available: <https://www.pololu.com/product/2464>.
203. IR Distance Sensor GP2Y0A41SK0F. [Type of medium]. Available: <https://www.smart-prototyping.com/Sharp-IR-Distance-Sensor-GP2Y0A41SK0F-4-30-cm>.
204. Adafruit 1918 Analog UV Light Sensor Breakout - GUVVA-S12SD. [Type of medium]. Available: <https://www.fabtolab.com/ada-analog-uv-light-sensor-breakout>.
205. Adafruit 1918 Analog UV Light Sensor Breakout - GUVVA-S12SD. [Type of medium]. Available: <https://www.amazon.com/Adafruit-1918-Analog-Sensor-Breakout/dp/B00LEVTOB0>.
206. DYP-ME007Y TX (Serial output) Ultrasonic Sensor interfacing with Arduino / NodeMCU. [Type of medium]. Available: <https://www.setfirelabs.com/building-automation/dyp-me007y-tx-serial-output-ultrasonic-sensor-interfacing-with-arduino-nodemcu>.
207. ME007YS Waterproof Ultrasonic Sensor. [Type of medium]. Available: <https://opencircuit.shop/product/ME007YS-Waterproof-Ultrasonic-Sensor>.
208. SHT10 Temperature & Humidity Sensor Development Board Module. [Type of medium]. Available: <https://www.amazon.com/Temperature-Humidity-Sensor-Development-Module/dp/B07T3ZLTXB>.
209. Maxmoral 2-Pack RCWL-0516 Microwave Radar Sensor. [Type of medium]. Available: <https://www.amazon.com/Maxmoral-RCWL-0516-Microwave-Induction-Intelligent/dp/B07S221PB4>.

210. RCWL 0516 Microwave Motion Sensor Module Radar Sensor Body Induction Module. [Type of medium]. Available: <http://www.icstation.com/5pcs-rcwl-0516-microwave-motion-sensor-module-radar-sensor-body-induction-module-100ma-p-13196.html>.

211. RobotDyn - Line Tracking Sensor, for Robot and car DIY Arduino Projects. Digital Out. [Type of medium]. Available: <https://www.ubuy.co.id/en/product/2FM69QPO-robotdyn-5-pcs-line-tracking-sensor-for-robot-and-car-diy-arduino-projects-digital-out>.

212. Line tracking Sensor. For robotic and car DIY Arduino projects. Digital Out. [Type of medium]. Available: <https://www.aliexpress.com/item/32654587628.html>.

213. MAX44009 ambient light sensor example. [Type of medium]. Available: <http://www.esp32learning.com/code/esp32-and-max44009-ambient-light-sensor-example.php>.

214. GY-49 MAX44009 Ambient Light Sensor Module with 4P Pin Header Module. [Type of medium]. Available: <https://www.amazon.com/MAX44009-Ambient-Sensor-Module-Header/dp/B07MYNKKPJ>.

215. Interfacing MAX30102 Pulse Oximeter and Heart Rate Sensor. [Type of medium]. Available: <https://lastminuteengineers.com/max30102-pulse-oximeter-heart-rate-sensor-arduino-tutorial/>.

216. Pulse Oximeter And Heart Rate Sensor (MAX30102). [Type of medium]. Available: <https://www.smart-prototyping.com/Pulse-Oximeter-and-Heart-Rate-Sensor-MAX30102>.

217. HiLetgo SHT31-D Temperature and Humidity Sensor Breakout Digital Output Temperature and Humidity Sensor Module IIC I2C. [Type of medium]. Available: <https://www.amazon.com/HiLetgo-Temperature-Humidity-Interface-GY-SHT31-D/dp/B07ZSZW92J>.

218. SHT31 Temperature SHT31-D Humidity Sensor Module Microcontroller IIC I2C Breakout Weather 3V 5V. [Type of medium]. Available: <https://www.aliexpress.com/i/32954460744.html>

219. MAX471 Ampere Metering Module (0-3A). [Type of medium]. Available: <https://www.robotistan.com/max471-ampere-metering-module-0-3a>
220. MAX471 3 ampere Current Sensor Module. [Type of medium]. Available: <https://www.dnatechindia.com/GY-471-max-471-3-ampere-DC-current-sensor-module.html>
221. GUV A-S12SD. [Type of medium]. Available: <https://www.digikey.com/en/products/detail/genicom-co-ltd/GUVA-S12SD/9960951>
222. GUV A-S12SD UV Sensor Module & Circuit. [Type of medium]. Available: <https://www.electroschematics.com/guva-s12sd-uv-sensor-module-circuit/>
223. uxcell IR Flame Sensor Module Detector 4 Pin Temperature Detecting. [Type of medium]. Available: <https://www.amazon.com/uxcell-Detector-Temperature-Detecting-Arduino/dp/B07QSK4JL>
224. Infrared IR Flame Sensor Detector Fire Detection Module For Arduino 4 PIN. [Type of medium]. Available: <https://shopee.com.my/Infrared-IR-Flame-Sensor-Detector-Fire-Detection-Module-For-Arduino-4-PIN-i.132528683.2034009850>
225. GY-530 VL53L0X Laser Ranging Sensor Time-of-Flight (ToF). [Type of medium]. Available: <https://www.amazon.com/VL53L0X-Ranging-Distance-Measurement-Communication/dp/B07KDQ4XQ4>.
226. GY-530 / VL53L0X Time of Flight (ToF) laser ranging sensor - Inline C. [Type of medium]. Available: <https://www.b4x.com/android/forum/threads/gy-530-vl53l0x-time-of-flight-tof-laser-ranging-sensor-inline-c.82124/>.
227. Rgb And Gesture Sensor - Apds-9960 - Proximity Detection Sensor Module. [Type of medium]. Available: <https://www.indiamart.com/proddetail/rgb-and-gesture-sensor-apds-9960-proximity-detection-sensor-module-23642760455.html>

228. NOYITO APDS-9960 APDS9960 RGB Gesture Sensor Module Hand Gesture Recognition Moving Direction Ambient Light RGB Proximity Sensor Module Infrared Move Sensor. [Type of medium]. Available: <https://www.amazon.com/NOYITO-APDS-9960-Recognition-Direction-Proximity/dp/B07PMMZZ26>.

229. LM393 Sound Sensor Module. [Type of medium]. Available: <https://www.circuituncle.com/product/lm393-sound-sensor-module/>

230. Sound Detection Sensor Module. [Type of medium]. Available: <https://xcraft.com.ua/sound-detection-sensor-module>

231. Comimark 1Pcs VL6180 VL6180X Range Finder Optical Ranging Sensor Module for Arduino I2C. [Type of medium]. Available: <https://www.amazon.com/Comimark-VL6180X-Optical-Ranging-Arduino/dp/B07XP27K3F>

232. VL6180 VL6180X IR Proximity Sensor Range Finder Optical Ranging Sensor Module Ambient Light Sensor IIC I2C Gesture Recognition. [Type of medium]. Available: <https://www.aliexpress.com/i/32965005851.html>

233. DHT22 Digital Temperature and Humidity Sensor Module. [Type of medium]. Available: <https://www.robotistan.com/dht22-temp-humidity-sensor-module>

234. DHT22 Temperature and Humidity Sensor. [Type of medium]. Available: <https://quartzcomponents.com/products/dht22-temprature-and-humidity-sensor-module>

235. ACS712 Current Sensor AC/DC. [Type of medium]. Available: <https://www.direnc.net/acs712-akim-sensoru-30a-30a-modul-en>

236. ACS712 Current Sensor Module AC / DC + 5A / -5A. [Type of medium]. Available: <https://www.direnc.net/acs712-akim-sensoru-5a-5a-modul-en>

237. ACS714 5A 20A 30A 5V Isolate Current Sensor Breakout Board Filter Resistance Capacitor Hall Effect Sensor Module Replace ACS712. [Type of medium]. Available: <https://www.aliexpress.com/item/32867262125.html>

238. Laser Sensor Detector Module Receiver Transmitter for Arduino AVR PIC. [Type of medium]. Available: <https://www.amazon.com/Detector-Transmitter-Application-detection-Obstacle-avoiding/dp/B00VUPKP4A>

239. Diffuse reflection laser sensor module black and white line tracking module. [Type of medium]. Available: <https://www.ebay.com/itm/254302615914>

240. IR Infrared Slotted Optical Speed Measuring Sensor Detection Optocoupler Module For Motor Test. [Type of medium]. Available: <https://www.aliexpress.com/item/32809610073.html>

241. IR Infrared Speed Measuring Sensor Slotted Optical Optocoupler Detection Module. [Type of medium]. Available: https://www.qimiadvice.com/index.php?main_page=product_info&products_id=234737

242. PI CAMERA MODULE V2.1 - Raspberry Pi Camera v2.1. [Type of medium]. Available: <https://www.distrelec.de/en/raspberry-pi-camera-v2-raspberry-pi-pi-camera-module-v2/p/30134462>

243. Raspberry Pi Camera Board V2 8MP, Adafruit. [Type of medium]. Available: <https://www.distrelec.biz/en/raspberry-pi-camera-board-v2-8mp-adafruit-3099/p/30133603>

244. Raspberry Pi 3 Model B+. The final revision of our third-generation single-board computer. [Type of medium]. Available: <https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/>

245. Raspberry Pi 3 Model B. [Type of medium]. Available: <https://docs.rs-online.com/44ed/0900766b8168e195.pdf>

246. Raspberry Pi 3. [Type of medium]. Available: <https://components101.com/microcontrollers/raspberry-pi-3-pinout-features-datasheet>

247. Raspberry Pi power limit. [Type of medium]. Available: <https://qastack.com.ua/raspberrypi/51615/raspberry-pi-power-limitations>

248. GPIO. [Type of medium]. Available: <https://www.raspberrypi.org/documentation/usage/gpio/>.

249. Using I2C as a Serial port?. [Type of medium]. Available: <https://forum.arduino.cc/t/using-i2c-as-a-serial-port/411381>

250. I2C Datasheet. [Type of medium]. Available: https://category.alldatasheet.com/index.jsp?sSearchword=I2c&gclid=Cj0KCQjwr-SSBhC9ARIsANhzu15shd4gXN3pCPRtzip0g9-CDjFaGLyUoM9XfU796uINkg6RKX AoeE3UaAljJEALw_wcB

251. Understanding the I2C Bus. [Type of medium]. Available: <https://www.ti.com/lit/pdf/slva704>

252. Introduction to SPI Interface. [Type of medium]. Available: <https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html>

253. Serial Peripheral Interface (SPI). [Type of medium]. Available: <https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all>

254. UART: A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter. [Type of medium]. Available: <https://www.analog.com/en/analog-dialogue/articles/uart-a-hardware-communication-protocol.html>

255. UARTs. [Type of medium]. Available: <https://learn.sparkfun.com/tutorials/serial-communication/UARTs>

256. Principle of the UART interface. [Type of medium]. Available: <https://911electronic.com/uart-serial-communication-interface/>

257. Modbus Application Protocol Specification V1.1b. [Type of medium]. Available: https://modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf

258. Modbus.org Documentation. [Type of medium]. Available: https://ctlsys.com/support/modbus-org_documentation/

259. AT Commands Reference Guide. [Type of medium]. Available: https://www.sparkfun.com/datasheets/Cellular%20Modules/AT_Commands_Reference_Guide_r0.pdf

260. Point-to-Point Protocol (PPP) – Tutorialspoint. [Type of medium]. Available: <https://www.tutorialspoint.com/point-to-point-protocol-PPP>

261. What is PPP (Point-to-Point Protocol) and How Does it Work?. [Type of medium]. Available:<https://www.techtarget.com/searchnetworking/definition/PPP>
262. IrDA (Infrared Data Association) – GeeksforGeeks. [Type of medium]. Available:<https://www.geeksforgeeks.org/irda-infrared-data-association/>
263. Infrared Data Association (IrDA) - Network Encyclopedia. [Type of medium]. Available: <https://networkencyclopedia.com/infrared-data-association-irda/>
264. AIROC™ HCI UART Control Protocol - Infineon Technologies. [Type of medium]. Available:<https://www.cypress.com/file/462731/download>
265. Programmable parameters - PrimeCell UART (PL011). [Type of medium]. Available: <https://developer.arm.com/documentation/ddi0183/g/introduction/about-the-uart/programmable-parameters>
266. Mini UART eats some characters - Raspberry Pi Forums. [Type of medium]. Available: <https://forums.raspberrypi.com/viewtopic.php?t=233616>.
267. Operating system images – Raspberry Pi. [Type of medium]. Available:<https://www.raspberrypi.com/software/operating-systems/>
268. Android OS vs Raspbian. |What are the differences?. [Type of medium]. Available:<https://stackshare.io/stackups/android-os-vs-raspbian>
269. How to install Windows 10 IoT Core on Raspberry Pi 3. [Type of medium]. Available: <https://www.windowcentral.com/how-install-windows-10-iot-raspberry-pi-3>
270. Installing Raspbian with NOOBS - Projects - Raspberry Pi. [Type of medium]. Available: <https://projects.raspberrypi.org/en/projects/noobs-install>
271. Introducing the New Out Of Box Software (NOOBS). [Type of medium]. Available:<https://www.raspberrypi.com/news/introducing-noobs/>
272. Raspberry Pi Documentation. [Type of medium]. Available:<https://www.raspberrypi.com/documentation/computers/os.html>
273. Scratch: Module 1 - Projects - Raspberry Pi. [Type of medium]. Available: <https://projects.raspberrypi.org/en/codeclub/scratch-module-1>

274. How to Install Java on Raspberry Pi | Linuxize. [Type of medium]. Available: <https://linuxize.com/post/install-java-on-raspberry-pi/>
275. Best Programming IDEs for Raspberry Pi - Linux Hint. [Type of medium]. Available: <https://linuxhint.com/best-programming-ide-raspberry-pi/>
276. Mathematica no longer available for the Raspberry Pi?. [Type of medium]. Available: <https://community.wolfram.com/groups/-/m/t/1511397>
277. Installing LibreOffice on the Raspberry Pi - Pi My Life Up. [Type of medium]. Available: <https://pimylifeup.com/raspberrypi-libreoffice/>
278. How to remotely access the Desktop of your Raspberry Pi. [Type of medium]. Available: <https://magpi.raspberrypi.com/articles/vnc-raspberry-pi>
279. Chen, Pei-Jarn, Tian-Hao Hu, and Ming-Shyan Wang. (2022). Raspberry Pi-Based Sleep Posture Recognition System Using AIoT Technique. *Healthcare*, Volume 10, No. 3: 513. DOI: 10.3390/healthcare10030513
280. Agghey, Abel Z., Lunodzo J. Mwinuka, Sanket M. Pandhare, Mussa A. Dida, and Jema D. Ndibwile. (2021). "Detection of Username Enumeration Attack on SSH Protocol: Machine Learning Approach" *Symmetry* 13, no. 11: 2192. DOI: 10.3390/sym13112192
281. The Linux command line for beginners | Ubuntu. [Type of medium]. Available: <https://ubuntu.com/tutorials/command-line-for-beginners>
282. Alex Bradbury. [Type of medium]. Available: <https://asbradbury.org/>
283. Apeksha P Kulkarni; Vishwanath P Baligar. (2020). Real Time Vehicle Detection, Tracking and Counting Using Raspberry-Pi. *The 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*. 5-7 March. Bangalore, India. DOI: 10.1109/ICIMIA48430.2020.9074944
284. TightVNC: VNC-Compatible Free Remote Control. [Type of medium]. Available: <https://www.tightvnc.com/>
285. What is DHCP and why is it important? - EfficientIP. [Type of medium]. Available: <https://www.efficientip.com/what-is-dhcp-and-why-is-it-important/>

286. What is the Difference Between Static and Dynamic IP Address?. [Type of medium]. Available: <https://whatismyipaddress.com/dynamic-static>
287. The Remote Control and Access Solution. [Type of medium]. Available: <https://www.teamviewer.com/en-us/products/teamviewer/>
288. Luis Gomes; Zita A. Vale; Juan Manuel Corchado. (2020). Multi-Agent Microgrid Management System for Single-Board Computers: A Case Study on Peer-to-Peer Energy Trading. *IEEE Access*. Volume: 8, Page(s): 64169 – 64183. DOI: 10.1109/ACCESS.2020.2985254
289. Juca, Sandro. Aplicações Práticas de sistemas embarcados Linux utilizando Raspberry Pi [recurso eletrônico] / Sandro Jucá e Renata Pereira. 1ª ed. - Rio de Janeiro: PoD, 2018.
290. How to Setup a Raspberry Pi Samba Server - Pi My Life Up.[Type of medium]. Available: <https://pimylifeup.com/raspberry-pi-samba/>
291. Samba on Raspberry Pi Guide - A To Z - KaliTut .[Type of medium]. Available: <https://kalitut.com/samba-on-raspberry-pi/>
292. CIFS vs SMB: What's the Difference? - Varonis. [Type of medium]. Available: <https://www.varonis.com/blog/cifs-vs-smb>
293. NetDrive - Mount your storage as local drive. [Type of medium]. Available: <https://www.netdrive.net/>
294. Limitations & Known Issues - Help - NetDrive. [Type of medium]. Available: <https://www.netdrive.net/wiki/netdrive/limitations/>
295. Linux permissions: SUID, SGID, and sticky bit | Enable Sysadmin. [Type of medium]. Available: <https://www.redhat.com/sysadmin/suid-sgid-sticky-bit>
296. Raspberry Pi Documentation - Using Linux. [Type of medium]. Available: https://www.raspberrypi.com/documentation/computers/using_linux.html
297. Win32 Disk Imager. A Windows tool for writing images to USB sticks or SD/CF cards. [Type of medium]. Available: <https://sourceforge.net/projects/win32diskimager/>

298. Disk Utility User Guide. [Type of medium]. Available: <https://support.apple.com/guide/disk-utility/welcome/mac>.
299. Python Source Releases. [Type of medium]. Available: <https://www.python.org/downloads/source/>
300. Thinesh Prathaban, Weilynn Thean, and Mohd Ilyas. (2019). A vision-based home security system using OpenCV on Raspberry Pi 3. *AIP Conference Proceedings*. Volume 2173, Issue 1. DOI: 10.1063/1.5133928
301. Ashwin Pajankar. Raspberry Pi Computer Vision Programming: Design and implement computer vision applications with Raspberry Pi, Open CV and Python 3. Second Edition. *Published by Packet Publishing Ltd*. P.296.
302. Bandopadhyay, D., Jha, V., Bandyopadhyay, A., Roy, P., Halder, R., Majhi, S. (2022). Automated People Monitoring System Using OpenCV and Raspberry Pi. In: Fong, S., Dey, N., Joshi, A. (eds) *ICT Analysis and Applications. Lecture Notes in Networks and Systems*, vol 314. Springer, Singapore. DOI:10.1007/978-981-16-5655-2_86
303. Setting up Raspberry Pi for Computer Vision (installing OpenCV). [Type of medium]. Available: <https://towardsdatascience.com/setting-up-raspberry-pi-for-computer-vision-installing-opencv-e0f973d38819>
304. Shalini K; Abhishek Kumar Srivastava; Surendra Allam; Dilip Lilaramani. (2021). Comparative analysis on Deep Convolution Neural Network models using Pytorch and OpenCV DNN frameworks for identifying optimum fruit detection solution on RISC-V architecture. *The IEEE Mysore Sub Section International Conference (MysuruCon)*. 24-25 Oct. Hassan, India. DOI: 10.1109/MysuruCon52639.2021.9641594
305. Singh, H. (2019). Advanced Image Processing Using OpenCV. In: *Practical Machine Learning and Image Processing*. Apress, Berkeley, CA. DOI:10.1007/978-1-4842-4149-3_4

306. Naman Gupta; Purushottam Sharma; Vikas Deep; Vinod Kumar Shukla. (2020). *The 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*. 4-5 June. Noida, India. DOI: 10.1109/ICRITO48877.2020.9197936.
307. Installation in Linux - OpenCV documentation. [Type of medium]. Available: https://docs.opencv.org/4.x/d7/d9f/tutorial_linux_install.html
308. Compile OpenCV 3.4.0 - UDOO Quad/Dual Docs. [Type of medium]. Available: https://www.udoo.org/docs/Cookbook_Linux/Compile_OpenCV_3.4.0.html
309. How to include libgtk2.0-dev and pkg-config in cmake when [Type of medium]. Available: <https://stackoverflow.com/questions/42843316/how-to-include-libgtk2-0-dev-and-pkg-config-in-cmake-when-installing-opencv-on-u>
310. Install OpenCV-Python in Ubuntu. [Type of medium]. Available: https://docs.opencv.org/4.x/d2/de6/tutorial_py_setup_in_ubuntu.html
311. Unable to locate package libjasper-dev - opencv - Stack [Type of medium]. Available: <https://www.codetd.com/en/article/12223433>
312. How to install OpenCV on Linux (Ubuntu) - gists. [Type of medium]. Available: <https://gist.github.com/8382707>
313. Install latest OpenCV on Ubuntu - Tutorial Kart. [Type of medium]. Available: <https://www.tutorialkart.com/opencv/install-latest-opencv-ubuntu-16/>
314. Install Opencv GitBook. [Type of medium]. Available: <https://robots.uc3m.es/installation-guides/install-opencv.html>
315. Can not create x264 VideoWriter Issue #81 opencv - GitHub. [Type of medium]. Available: <https://github.com/opencv/opencv-python/issues/81>
316. Install latest OpenCV on Ubuntu - Tutorial Kart. [Type of medium]. Available: <https://www.tutorialkart.com/opencv/install-latest-opencv-ubuntu-16/>
317. Sage-git/forcv: A Fortran wrapper for OpenCV - GitHub. [Type of medium]. Available: <https://github.com/sage-git/forcv>

318. Steps To install Opencv on a Raspberry Pi - Discover gists. [Type of medium]. Available:<https://gist.github.com/preetjdp/46c3fdc77a1c5d266949dd47027670cc>
319. Sullivan et al., (2019). PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK). *Journal of Open Source Software*, 4(37), 1450. DOI: 10.21105/joss.01450
320. Aashish Chaudhary; Sankhesh J. Jhaveri; Alvaro Sanchez; Lisa S. Avila; Kenneth M. Martin; Allison Vacanti; Marcus D. Hanwell; et al., (2019). Cross-Platform Ubiquitous Volume Rendering Using Programmable Shaders in VTK for Scientific and Medical Visualization. *IEEE Computer Graphics and Applications*. Volume: 39, Issue: 1. Page(s): 26 – 43. DOI: 10.1109/MCG.2018.2880818
321. Sullivan et al., (2019). PVGeo: an open-source Python package for geoscientific visualization in VTK and ParaView. *Journal of Open Source Software*, 4(38), 1451. DOI:10.21105/joss.01451
322. Raspberry-gpio-python Files. A Python module to control the GPIO on a Raspberry Pi. [Type of medium]. Available: <https://sourceforge.net/projects/raspberry-gpio-python/files/?%20Source%20=%20navbar>.
323. Strickland, J.R. (2018). Meet WiringPi. In: *Raspberry Pi for Arduino Users*. Apress, Berkeley, CA. DOI:10.1007/978-1-4842-3414-3_5
324. Bräunl, T. (2022). Raspberry Pi. In *Embedded Robotics*. Springer, Singapore. DOI:10.1007/978-981-16-0804-9_4
325. Wang, SH., Cheng, SW., Huang, CC.(. (2019). Puyuma: Linux-Based RTOS Experimental Platform for Constructing Self-driving Miniature Vehicles. In: Arai, K., Kapoor, S., Bhatia, R. (eds) *Intelligent Computing. SAI 2018. Advances in Intelligent Systems and Computing*, vol 858. Springer, Cham. DOI:10.1007/978-3-030-01174-1_75

326. Raspberry Pi Distance Sensor using the HC-SR04. [Type of medium]. Available: <https://pimylifeup.com/raspberry-pi-distance-sensor/>.
327. Raspberry Pi Humidity Sensor using the DHT22. [Type of medium]. Available: <https://pimylifeup.com/raspberry-pi-humidity-sensor-dht22/>
328. M F Wicaksono, Syahrul, M D Rahmatya and M A F Rahman. (2020). Raspberry Pi-Based Solar System Learning Media. *IOP Conference Series: Materials Science and Engineering*. Volume 879, P.365-378.
329. Susanti, D., & Syarif Abdurrohman, M. (2020). Rancang Bangun Alat Pemindai Dokumen Menggunakan Raspberry Pi. *Infotech Journal*, 6(1), 44-48. DOI:10.31949/infotech.v6i1.506
330. Aashika Prasad, M. Gowtham, S. Mohanraman, M. Suresh. (2020). Automatic Sorting Machine. *International Research Journal of Multidisciplinary Technovation* Volume 20(1): 7 -12. DOI: 10.34256/irjmt2102
331. Jun Yang; Han Wu; Liang Hu; Shihua Li. (2018). Robust Predictive Speed Regulation of Converter-Driven DC Motors via a Discrete-Time Reduced-Order GPIO. *IEEE Transactions on Industrial Electronics*. Volume: 66, Issue: 10. Page(s): 7893 – 7903. DOI: 10.1109/TIE.2018.2878119
332. Sai Munikoti; Laya Das; Balasubramaniam Natarajan; Babji Srinivasan. (2019). Data-Driven Approaches for Diagnosis of Incipient Faults in DC Motors. *IEEE Transactions on Industrial Informatics*. Volume: 15, Issue: 9, Page(s): 5299 – 5308. DOI: 10.1109/TII.2019.2895132.
333. Tuanjie Li; Jiaying Zhou. High-Stability Position-Sensorless Control Method for Brushless DC Motors at Low Speed. *IEEE Transactions on Power Electronics*. Volume: 34, Issue: 5, Page(s): 4895 – 4903. DOI: 10.1109/TPEL.2018.2863735
334. Jesus Lopez-Gomez; M. Aurora D. Vargas-Treviño; Sergio Vergara-Limon; Marciano Vargas-Treviño; Jaime Gutierrez-Gutierrez; A. D. Palo. (2020). Influence of PWM Torque Control Frequency in DC Motors by Means of an Optimum Design Method. *IEEE Access*, Volume: 8, Page(s): 80691 – 80706. DOI: 10.1109/ACCESS.2020.2990158.

335. Pranav Adarsh; Pratibha Rathi; Manoj Kumar. (2020). YOLO v3-Tiny: Object Detection and Recognition using one stage improved model. *The 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*. 6-7 March. Coimbatore, India. DOI: 10.1109/ICACCS48705.2020.9074315
336. Kortli, Yassin, Maher Jridi, Ayman Al Falou, and Mohamed Atri. (2020). Face Recognition Systems: A Survey" *Sensors* 20, no. 2: 342. DOI:10.3390/s20020342
337. Mate Krišto; Marina Ivasic-Kos; Miran Pobar. (2020). Thermal Object Detection in Difficult Weather Conditions Using YOLO. *IEEE Access*. Volume: 8, Page(s): 125459 – 125476. DOI: 10.1109/ACCESS.2020.3007481
338. Sudha Sharma; Mayank Bhatt; Pratyush Sharma. (2020). Face Recognition System Using Machine Learning Algorithm. *The 5th International Conference on Communication and Electronics Systems (ICCES)*. 10-12 June. Coimbatore, India. DOI: 10.1109/ICCES48766.2020.9137850
339. Dimity Miller; Lachlan Nicholson; Feras Dayoub; Niko Sünderhauf. (2018). Dropout Sampling for Robust Object Detection in Open-Set Conditions. *The IEEE International Conference on Robotics and Automation (ICRA)*. 21-25 May. Brisbane, QLD, Australia. DOI: 10.1109/ICRA.2018.8460700
340. Francisco Pérez-Hernández, Siham Tabik, Alberto Lamas, Roberto Olmos, HamidoFujita, Francisco Herrera. (2020). Object Detection Binary Classifiers methodology based on deep learning to identify small objects handled similarly: Application in video surveillance. *Knowledge-Based Systems*. Volume 194. DOI: 10.1016/j.knosys.2020.105590
341. Raspberry Pi Camera Pinout – Arducam. [Type of medium]. Available: <https://www.arducam.com/raspberry-pi-camera-pinout/>

342. Tripathi, A., Pandey, A.B., Singh, A.K., Malik, P., Singh, K.K., Vashist, P.C. (2021). IoT for Smart Automation and Robot. In: Singh, K.K., Nayyar, A., Tanwar, S., Abouhawwash, M. (eds) Emergence of Cyber Physical System and IoT in Smart Automation and Robotics. Advances in Science, Technology & Innovation. Springer, Cham. DOI:10.1007/978-3-030-66222-6_13

343. Ian Edwards. (2018). Search like a robot: Developing targeted search algorithms. *Australian Law Librarian*. Volume:26, Issue:2, Page Range:104-109. ISSN:1039-6616.

344. Ye Yuan; Guang-Zhong Cao; Aibin Zhu; Xing Lyu; Yulong Wang. (2020). Communication Scheme of Cloud Platform for the Lower Limb Exoskeleton Rehabilitation Robot. *The 17th International Conference on Ubiquitous Robots (UR)*. 22-26 June. Kyoto, Japan. DOI: 10.1109/UR49135.2020.9144989

345. C Ramasamy Sankar Ram, S Ravimaran, R Santhana Krishnan. (2020). Internet of Green Things with autonomous wireless wheel robots against green houses and farms. *International Journal of Distributed Sensor Networks* .Volume: 16 issue: 6. DOI: 10.1177/1550147720923477.

346. Vega, Julio, and José M. Cañas. (2019). PyBoKids: An Innovative Python-Based Educational Framework Using Real and Simulated Arduino Robots. *Electronics* 8, no. 8: 899. DOI:10.3390/electronics8080899

347. Xiaoqi Wang, Xing Liu, Lerui Chen, Heyu Hu. (2021). Deep-learning damped least squares method for inverse kinematics of redundant robots. *Measurement*. Volume 171. DOI: 10.1016/j.measurement.2020.108821

348. Ashish Sharma; Paras Chugh; Rohan Aggarwal; Harshit Garg. (2021). Shell Shoveling Using Socket Programming. *The International Conference on Industrial Electronics Research and Applications (ICIERA)*. 22-24 Dec. New Delhi, India. DOI: 10.1109/ICIERA53202.2021.9726756.

349. Learn C Programming – Programiz. [Type of medium]. Available: <https://www.programiz.com/c-programming/examples>.
350. Download .NET Framework 4.7 | Free official downloads. [Type of medium]. Available: <https://dotnet.microsoft.com/en-us/download/dotnet-framework/net47>
351. HTML: HyperText Markup Language - MDN Web Docs. [Type of medium]. Available: <https://developer.mozilla.org/en-US/docs/Web/HTML>
352. Operating system images – Raspberry Pi. [Type of medium]. Available: <https://www.raspberrypi.com/software/operating-systems/>
353. SSH Protocol – Secure Remote Login and File Transfer. [Type of medium]. Available: <https://www.ssh.com/academy/ssh/protocol>
354. Nikola Tomasevic, Nikola Gvozdenovic, Sanja Vranes. (2020). An overview and comparison of supervised data mining techniques for student exam performance prediction. *Computers & Education*. Volume 143. DOI: 10.1016/j.compedu.2019.103676
355. Mohammad Shafenoor Amin, Yin Kia Chiam, Kasturi Dewi Varathan. (2019). Identification of significant features and data mining techniques in predicting heart disease. *Telematics and Informatics*. Volume 36, Pages 82-93. DOI: 10.1016/j.tele.2018.11.007
356. Tarawneh, M., Embarak, O. (2019). Hybrid Approach for Heart Disease Prediction Using Data Mining Techniques. In: Barolli, L., Xhafa, F., Khan, Z., Odhabi, H. (eds) *Advances in Internet, Data and Web Technologies. EIDWT 2019. Lecture Notes on Data Engineering and Communications Technologies*, vol 29. Springer, Cham. DOI:10.1007/978-3-030-12839-5_41
357. Sultana, Z., Nahar, L., Basnin, N., Hossain, M.S. (2021). Inference and Learning Methodology of Belief Rule Based Expert System to Assess Chikungunya. In: Mahmud, M., Kaiser, M.S., Kasabov, N., Iftekharuddin, K., Zhong, N. (eds) *Applied Intelligence and Informatics. AII 2021. Communications in Computer and Information Science*, vol 1435. Springer, Cham. DOI: 10.1007/978-3-030-82269-9_1

358. Nassr, Mohammed S. and Abu Naser, Samy S., (2018). Knowledge Based System for Diagnosing Pineapple Diseases. *International Journal of Academic Pedagogical Research (IJAPR)*, 2(7), P:12-19. Available at SSRN: <https://ssrn.com/abstract=3219802>
359. Yanqing Duan, John S. Edwards, Yogesh K Dwivedi. (2019). Artificial intelligence for decision making in the era of Big Data – evolution, challenges and research agenda. *International Journal of Information Management*. Volume 48, Pages 63-71. DOI: 10.1016/j.ijinfomgt.2019.01.021.
360. Alzubaidi, L., Zhang, J., Humaidi, A.J. et al. (2021) Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *J Big Data* 8, 53. DOI:10.1186/s40537-021-00444-8
361. Yuzhu Ji, Haijun Zhang, Zhao Zhang, Ming Liu. (2021). CNN-based encoder-decoder networks for salient object detection: A comprehensive review and recent advances. *Information Sciences*, Volume 546, Pages 835-857. DOI: 10.1016/j.ins.2020.09.003
362. Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, Alexei A. Efros. (2020). CNN-Generated Images Are Surprisingly Easy to Spot... for Now. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 8695-8704.
363. Yalçın, O.G. (2021). Convolutional Neural Networks. In: Applied Neural Networks with TensorFlow 2. *Apress, Berkeley, CA*. DOI:10.1007/978-1-4842-6513-0_7
364. Chechliński, Ł., Siemiątkowska, B., Majewski, M. (2018). A System for Weeds and Crops Identification Based on Convolutional Neural Network. In: Szewczyk, R., Zieliński, C., Kaliczyńska, M. (eds) Automation 2018. AUTOMATION 2018. *Advances in Intelligent Systems and Computing*, vol 743. Springer, Cham. DOI:10.1007/978-3-319-77179-3_18

365. Yaqub, Muhammad, Jinchao Feng, M. S. Zia, Kaleem Arshid, Kebin Jia, Zaka U. Rehman, and Atif Mehmood. (2020). State-of-the-Art CNN Optimizer for Brain Tumor Segmentation in Magnetic Resonance Images. *Brain Sciences*, 10, no. 7: 427. DOI:10.3390/brainsci10070427
366. Colors RGB - W3Schools. [Type of medium]. Available:https://www.w3schools.com/colors/colors_rgb.asp
367. #NurderHSV: Alle Informationen über den HSV | HSV.de. [Type of medium]. Available: <https://www.hsv.de/>
368. CMYK to Pantone Color Converter - URL-Decode. [Type of medium]. Available:<https://url-decode.com/tool/cmyk-to-pantone>
369. Yong Zhu, Guangpeng Li, Rui Wang, Shengnan Tang, Hong Sua Kai Cao. (2021). Intelligent fault diagnosis of hydraulic piston pump combining improved LeNet-5 and PSO hyperparameter optimization. *Applied Acoustics*. Volume 183. DOI: 10.1016/j.apacoust.2021.108336
370. Lu, S., Wang, SH. & Zhang, YD. (2021).Detection of abnormal brain in MRI via improved AlexNet and ELM optimized by chaotic bat algorithm. *Neural Comput & Applic* 33, 10799–10811 DOI:10.1007/s00521-020-05082-4
371. Zheng Wang, Xu Zheng, Dongyan Li, Helin Zhang, Yi Yang, Hongguang Pan. (2021). A VGGNet-like approach for classifying and segmenting coal dust particles with overlapping regions. *Computers in Industry*. DOI: 10.1016/j.compind.2021.103506
372. Lakshmipriya Balagourouchetty; Jayanthi K. Pragatheeswaran; Biju Pottakkat; G. Ramkumar. (2020). GoogLeNet-Based Ensemble FCNet Classifier for Focal Liver Lesion Diagnosis. *IEEE Journal of Biomedical and Health Informatics*. Volume: 24, Issue: 6, Page(s): 1686 – 1694. DOI: 10.1109/JBHI.2019.2942774
373. Wen, L., Li, X. & Gao, L. (2020).A transfer convolutional neural network for fault diagnosis based on ResNet-50. *Neural Comput & Applic* 32, 6111–6124.DOI:10.1007/s00521-019-04097-w

374. Arren Matthew C. Antioquia; Daniel Stanley Tan; Arnulfo Azcarraga; Wen-Huang Cheng; Kai-Lung Hua. (2018). ZipNet: ZFNet-level Accuracy with $48\times$ Fewer Parameters. *The 2018 IEEE Visual Communications and Image Processing (VCIP)*. 9-12 Dec. Taichung, Taiwan. DOI: 10.1109/VCIP.2018.8698672
375. Ping Chang, Guang Xu, Fubao Zhou, Benjamin Mullins, S.Abishek. (2019). Comparison of underground mine DPM simulation using discrete phase and continuous phase models. *Process Safety and Environmental Protection*. Volume 127, Pages 45-55. DOI: 10.1016/j.psep.2019.04.027
376. Shen Zheng; Yuxiong Wu; Shiyu Jiang; Changjie Lu; Gaurav Gupta. (2021). Deblur-YOLO: Real-Time Object Detection with Efficient Blind Motion Deblurring. *The International Joint Conference on Neural Networks (IJCNN)*. 18-22 July. Shenzhen, China. DOI: 10.1109/IJCNN52387.2021.9534352
377. Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., & Ren, D. (2020). Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07), 12993-13000. DOI:10.1609/aaai.v34i07.6999
378. Lu Chen, Panfeng Huang, Zhongjie Meng. (2019). Convolutional multi-grasp detection using grasp path for RGBD images. *Robotics and Autonomous Systems*. Volume 113, Pages 94-103. DOI: 10.1016/j.robot.2019.01.009
379. A.Iskhakova, A.Iskhakov, R.Meshcheryakov, E.Jharko. (2019). Method of Verification of Robotic Group Agents in the Conditions of Communication Facility Suppression. *IFAC-PapersOnLine*. Volume 52, Issue 13, Pages 1397-1402. DOI: 10.1016/j.ifacol.2019.11.394
380. Kyle Hunte; Jingang Yi (2019). Collaborative Object Manipulation Through Indirect Control of a Deformable Sheet by a Mobile Robotic Team. *The 15th International Conference on Automation Science and Engineering (CASE)*. 22-26 Aug. Vancouver, BC, Canada. DOI: 10.1109/COASE.2019.8843240

381. Peter Karkus, Shaojun Cai, David Hsu. (2021). Differentiable SLAM-Net: Learning Particle SLAM for Visual Navigation. *The Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2815-2825.
382. Jaradat, M.A., Bani-Salim, M. & Awad, F. (2018). A Highly-Maneuverable Demining Autonomous Robot: an Over-Actuated Design. *J Intell Robot Syst* 90, 65–80. DOI:10.1007/s10846-017-0654-y
383. Wanli Gu, Shuo Cai, Yunfeng Hua, Hui Zhang, Hong Chen. (2019). Trajectory planning and tracking control of a ground mobile robot:A reconstruction approach towards space vehicle. *ISA Transactions*. Volume 87, Pages 116-128. DOI: 10.1016/j.isatra.2018.11.019
384. Mark Debord; Wolfgang Hönig; Nora Ayanian. (2018). Trajectory Planning for Heterogeneous Robot Teams. *The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 1-5 Oct. Madrid, Spain. DOI: 10.1109/IROS.2018.8593876
385. Juncheng Li; Maopeng Ran; Lihua Xie. (2021). Efficient Trajectory Planning for Multiple Non-Holonomic Mobile Robots via Prioritized Trajectory Optimization. *IEEE Robotics and Automation Letters*. Volume: 6, Issue: 2, Page(s): 405 – 412. DOI: 10.1109/LRA.2020.3044834
386. Daniel Foad, Alifio Ghifari, Marchel Budi, Kusuma, Novita Hanafiah, Eric Gunawan. (2021). A Systematic Literature Review of A* Pathfinding. *Procedia Computer Science*. Volume 179, Pages 507-514. DOI: 10.1016/j.procs.2021.01.034
387. Stern, R. (2019). Multi-Agent Path Finding – An Overview. In: Osipov, G., Panov, A., Yakovlev, K. (eds) Artificial Intelligence. *Lecture Notes in Computer Science*, vol 11866. Springer, Cham. DOI:10.1007/978-3-030-33274-7_6
388. Charis Ntakoli, Dimitris K.Iakovidis. (2021). A swarm intelligence graph-based pathfinding algorithm (SIGPA) for multi-objective route planning. *Computers & Operations Research*. Volume 133. DOI: 10.1016/j.cor.2021.105358

389. Divya Agarwal, Pushpendra S. Bharti. (2018). A Review on Comparative Analysis of Path Planning and Collision Avoidance Algorithms. *World Academy of Science, Engineering and Technology. International Journal of Mechanical and Mechatronics Engineering*. Vol:12, No:6, P.608 – 624.
390. Diep Quoc Bao, Ivan Zelinka. (2019). Obstacle Avoidance for Swarm Robot Based on Self-Organizing Migrating Algorithm. *Procedia Computer Science*. Volume 150, Pages 425-432. DOI: 10.1016/j.procs.2019.02.073.
391. Xinyu Li, Zuhua Jiang, Yeqin Guan, Geng Li, Fuhua Wang. (2019). Fostering the transfer of empirical engineering knowledge under technological paradigm shift: An experimental study in conceptual design. *Advanced Engineering Informatics*. Volume 41, DOI: 10.1016/j.aei.2019.100927
392. Chaozheng Zhu, Ming He, Pan Chen, Kang Sun, Jinglei Wang and Qian Huang. (2018). Navigation for Indoor Robot: Straight Line Movement via Navigator. *Mathematical Problems in Engineering*. Article ID8419384. DOI: 10.1155/2018/8419384.
393. Al Ani, Z., Gujarathi, A.M., Vakili-Nezhaad, G.R. et al. (2020). Hybridization Approach Towards Improving the Performance of Evolutionary Algorithm. *Arab J Sci Eng* 45, 11065–11086. DOI:10.1007/s13369-020-04964-y
394. Sumathi, D. P. (2021). Efficient Multi Keyword Search and Retrieve Encrypted Cloud Data Using Iddfs and Fpgrowth Algorithm. *Annals of the Romanian Society for Cell Biology*, 25(6), 7786–7794.
395. Cristian Tatino; Nikolaos Pappas; Di Yuan. (2020). Multi-Robot Association-Path Planning in Millimeter-Wave Industrial Scenarios. *IEEE Networking Letters*, Volume: 2, Issue: 4. Page(s): 190 – 194. DOI: 10.1109/LNET.2020.3037741
396. Rafal Szczepanski; Tomasz Tarczewski. (2021). Global path planning for mobile robot based on Artificial Bee Colony and Dijkstra's algorithms. *The IEEE 19th International Power Electronics and Motion Control Conference (PEMC)*. 25-29 April. Gliwice, Poland. DOI: 10.1109/PEMC48073.2021.9432570.

397. Parimala, M., Broumi, S., Prakash, K. et al. (2021). Bellman–Ford algorithm for solving shortest path problem of a network under picture fuzzy environment. *Complex Intell. Syst.* 7, 2373–2381. DOI: 10.1007/s40747-021-00430-w
398. Khaing Khaing Wai. (2019). Analysis of RIP, EIGRP, and OSPF Routing Protocols in a Network. *International Journal of Trend in Scientific Research and Development (IJTSRD)*. Volume 3 Issue 5, pp.2484-2487. DOI: 10.31142/ijtsrd27928
399. Annu Lambora; Kunal Gupta; Kriti Chopra. (2019). Genetic Algorithm- A Literature Review. *The International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*. 14-16 Feb. DOI: 10.1109/COMITCon.2019.8862255
400. Shehab Abdulhabib Saeed Alzaeemi, Saratha Sathasivam, Muraly Velavan. (2020). Agent-based Modeling in doing Logic Programming in Fuzzy Hopfield Neural Network. *I.J. Modern Education and Computer Science*, 2021, 2, P.23-32. DOI: 10.5815/ijmecs.2021.02.03
401. He H, Shang Y, Yang X, Di Y, Lin J, Zhu Y, Zheng W, Zhao J, Ji M, Dong L, Deng N, Lei Y and Chai Z (2019) Constructing an Associative Memory System Using Spiking Neural Network. *Front. Neurosci.* 13:650. DOI: 10.3389/fnins.2019.00650
402. Robot Technology and Applications. Edit. Uilrich Rembold. *Taylor&Francis Grup*. 1990. P. 289
403. Habibian, S., Dadvar, M., Peykari, B. et al. (2021). Design and implementation of a maxi-sized mobile robot (Karo) for rescue missions. *Robomech J* 8, 1. DOI: 10.1186/s40648-020-00188-9
404. Fabio Ruggiero; Vincenzo Lippiello; Anibal Ollero. (2018). Aerial Manipulation: A Literature Review. *IEEE Robotics and Automation Letters*, Volume: 3, Issue: 3. Page(s): 1957 – 1964. DOI: 10.1109/LRA.2018.2808541

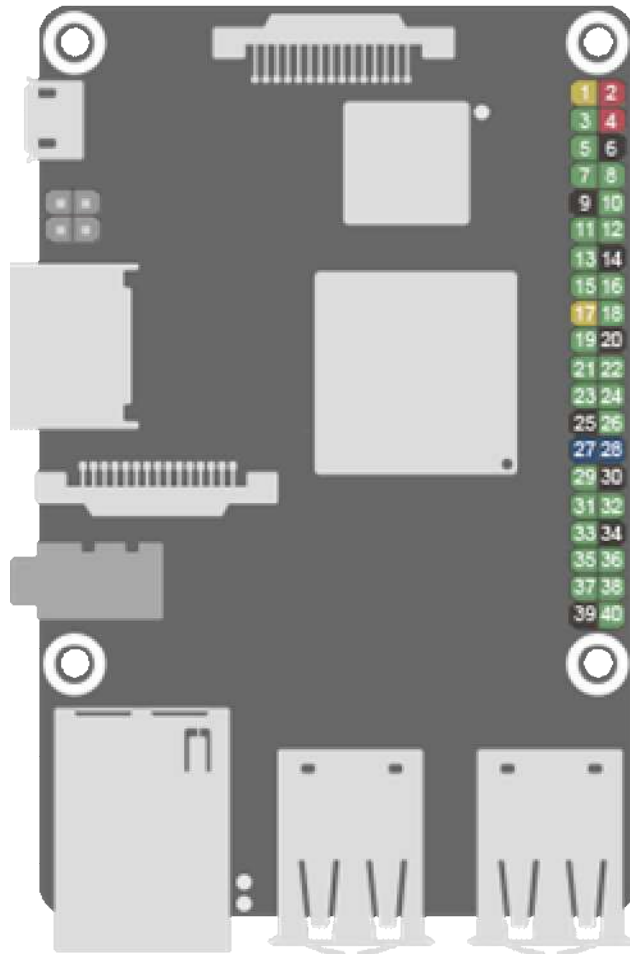
405. Tobias Klamt; Diego Rodriguez; Max Schwarz; Christian Lenz; Dmytro Pavlichenko; David Droeschel; Sven Behnke. (2018). Supervised Autonomous Locomotion and Manipulation for Disaster Response with a Centaur-Like Robot. *The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 1-5 Oct. Madrid, Spain. DOI: 10.1109/IROS.2018.8594509
406. ROS: Home. [Type of medium]. Available: <https://www.ros.org/>
407. What does UNIX like mean?.[Type of medium]. Available: <https://www.compuhoy.com/what-does-unix-like-mean/>
408. AQMD6010BLS E2 BLDC Motor Driver 9-60V 600W Brushless DC Current/Speed/Position. [Type of medium]. Available: <https://www.ebay.com/itm/284541239459>
409. Blog - What is a 18650 battery? - NKON.nl. [Type of medium]. Available: <https://www.nkon.nl/ru/blog/18650battery/>
410. MPU-9150 - InvenSense. [Type of medium]. Available:<https://invensense.tdk.com/products/motion-tracking/9-axis/mpu-9150-2/>
411. What is UDP? | Cloudflare. [Type of medium]. Available:<https://www.cloudflare.com/learning/ddos/glossary/user-datagram-protocol-udp/>
412. Bhookya, J., Jatoth, R.K. (2019). Optimal FOPID/PID controller parameters tuning for the AVR system based on sine–cosine-algorithm. *Evol. Intel.* 12, P.725–733. DOI:10.1007/s12065-019-00290-x
413. Interfacing DS18B20 1-Wire Digital Temperature Sensor.[Type of medium]. Available: <https://lastminuteengineers.com/ds18b20-arduino-tutorial/>
414. BML04PE (BML-S1G0-B7ED-M5EA-D0-S284) Absolute linear. [Type of medium]. Available:<https://www.balluff.com/en-th/products/BML04PE>
415. E18-D80NK IR Infrared Sensor Module for Transparent. [Type of medium]. Available:<https://www.amazon.in/E18-D80NK-Infrared-Sensor-Module-Transparent/dp/B01LYUSBTC>

416. DigChip IC database. [Type of medium]. Available: https://www.digchip.com/datasheets/parts/ho/parts_ho.php
417. Arduino Waterproof JSN-SR04T Ultrasonic Sensor. [Type of medium]. Available: <https://www.makerguides.com/jsn-sr04t-arduino-tutorial/>
418. STM32 32-bit Arm Cortex MCUs - STMicroelectronics. [Type of medium]. Available: <https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html>
419. AS5040 - 10-bit rotary position sensor with absolute. [Type of medium]. Available: <https://ams.com/en/as5040>
420. Depth Camera D455 - Intel RealSense. [Type of medium]. Available: <https://www.intelrealsense.com/depth-camera-d455/>
421. Intel-RealSense-D400-Series-Datasheet-June-2020. [Type of medium]. Available: <https://www.intelrealsense.com/wp-content/uploads/2020/06/Intel-RealSense-D400-Series-Datasheet-June-2020.pdf>
422. Pyojin Kim; Brian Coltin; H. Jin Kim. (2018). Low-Drift Visual Odometry in Structured Environments by Decoupling Rotational and Translational Motion. *The IEEE International Conference on Robotics and Automation (ICRA)*. 21-25 May. Brisbane, QLD, Australia. DOI: 10.1109/ICRA.2018.8463207
423. Sierra N. Young; Joshua M. Peschel. (2020). Review of Human–Machine Interfaces for Small Unmanned Systems With Robotic Manipulators. *IEEE Transactions on Human-Machine Systems*. Volume: 50, Issue: 2. Page(s): 131 – 143. DOI: 10.1109/THMS.2020.2969380
424. Simone A. Ludwig; Kaleb D. Burnham. (2018). Comparison of Euler Estimate using Extended Kalman Filter, Madgwick and Mahony on Quadcopter Flight Data. *The International Conference on Unmanned Aircraft Systems (ICUAS)*. 12-15 June. Dallas, TX, USA. DOI: 10.1109/ICUAS.2018.8453465

425. Cătălin-Cosmin Golban; Corvin-Petruț Cobârzan; Sergiu Nedevschi. (2021). Visual Odometry Drift Reduction Based on LiDAR Point Clouds Alignment. *The 17th International Conference on Intelligent Computer Communication and Processing (ICCP)*. 28-30 Oct. Cluj-Napoca, Romania. DOI: 10.1109/ICCP53602.2021.9733667

ДОДАТОК А

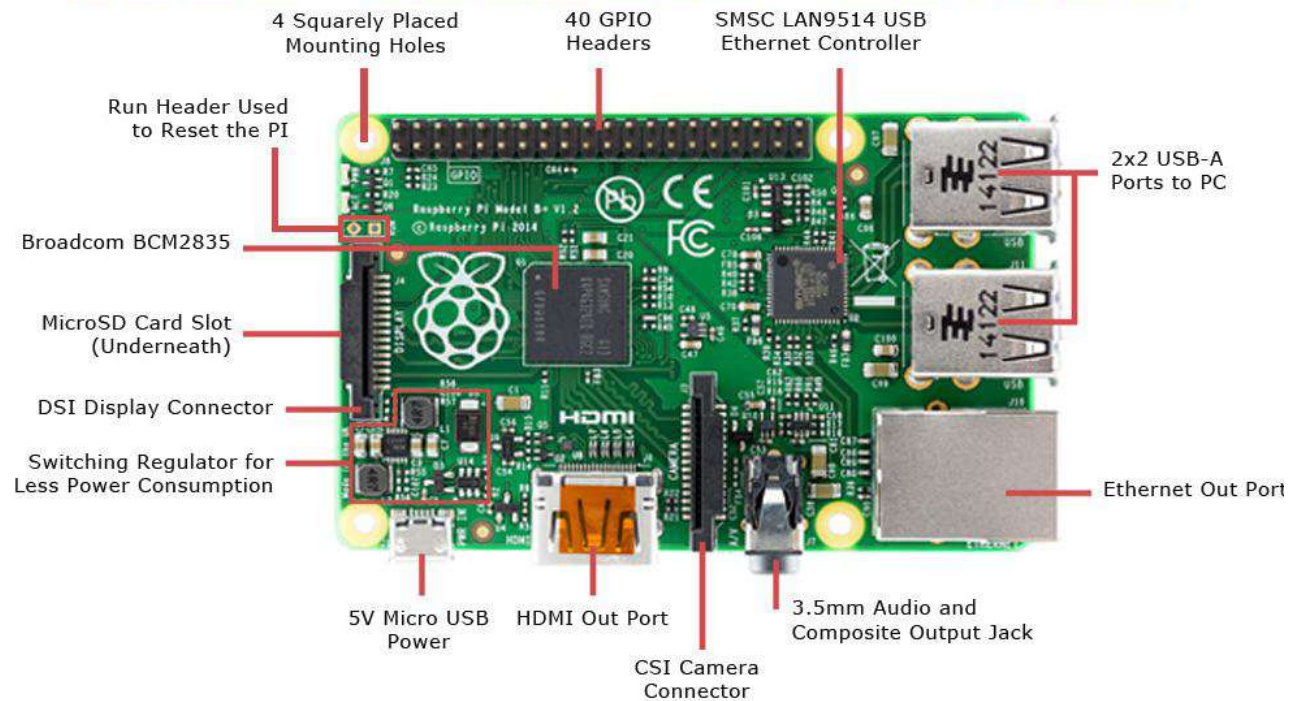
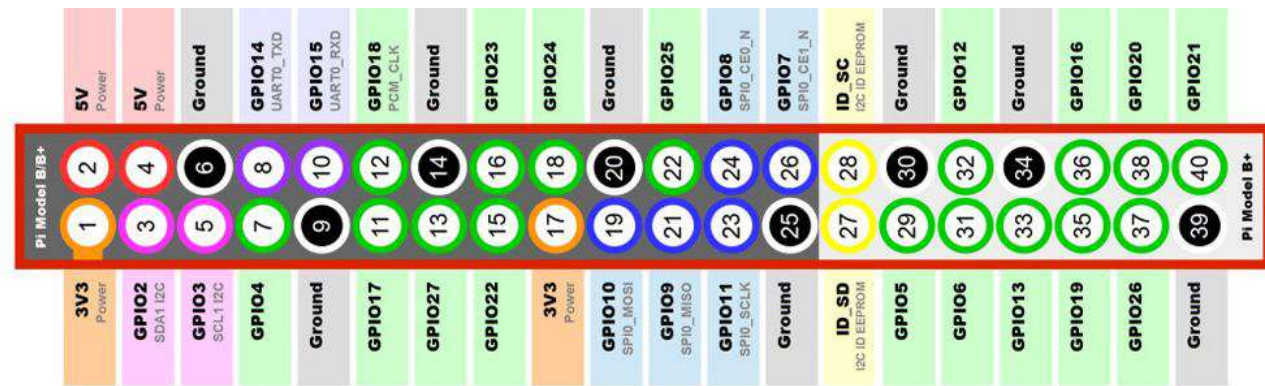
Розташування виводів Asus Tinker Board 2Гб на GPIO



1	VCC3.3V_IO	2	VCC5V_SYS
3	GP8A4_I2C1_SDA	4	VCC5V_SYS
5	GP8A5_I2C1_SCL	6	GND
7	GP0C1_CLKOUT	8	GP5B1_UART1TX
9	GND	10	GP5B0_UART1RX
11	GP5B4_SPI0CLK_UART4CTSN	12	GP6A0_PCM/I2S_CLK
13	GP5B6_SPI0_TXD_UART4TX	14	GND
15	GP5B7_SPI0_RXD_UART4RX	16	GP5B2_UART1CTSN
17	VCC3.3V_IO	18	GP5B3_UART1RTSN
19	GP8B1_SPI2TXD	20	GND
21	GP8B0_SPI2RXD	22	GP5C3
23	GP8A6_SPI2CLK	24	GP8A7_SPI2CSN0
25	GND	26	GP8A3_SPI2CSN1
27	GP7C1_I2C4_SDA	28	GP7C2_I2C4_SCL
29	GP5B5_SPI0CSN0_UART4RTSN	30	GND
31	GP5C0_SPI0CSN1	32	GP7C7_UART2TX_PWM3
33	GP7C6_UART2RX_PWM2	34	GND
35	GP6A1_PCM/I2S_FS	36	GP7A7_UART3RX
37	GP7B0_UART3TX	38	GP6A3_PCM/I2S_SDI
39	GND	40	GP6A4_PCM/I2S_SDO

ДОДАТОК Б

Розташування виводів Raspberry Pi 2 &3 на GPIO



ДОДАТОК В

Порівняльні характеристик основних моделі Raspberry

	Model A	Model B	Model A +	Model B+	Raspberry Pi 2	Raspberry Pi 3	Raspberry Pi Zero
							
1	2	3	4	5	6	7	8
SoC	Broadcom BCM2835				Broadcom BCM2836	Broadcom BCM2837	Broadcom BCM2835
CPU	700 MHz ARM11 ARM1176JZF-S core				4-ех ядерный 900MHz ARMv7 Cortex-A7	4-ех ядерный 1.2GHz 64-bit ARMv8 Cortex-A53	1000MHz ARM1176JZ-F c LE
GPU	Broadcom VideoCore IV, OpenGL ES 2.0,OpenVG 1080p30 H.264, 250 MHz					Broadcom VideoCore IV, OpenGL ES 2.0,OpenVG 1080p60 H.264, 400 MHz	Broadcom VideoCore IV
SDRAM	256MB	512MB*	256MB	512 MB	1024 MB		512 MB
USB порты	1 USB 2.0	2 USB 2.0	1 USB 2.0	4 USB 2.0			1 Micro USB OTG
Video Out.	Композитний відеовихід RCA HDMI (одночасне виведення зображення через RCA і HDMI не підтримується)		HDMI Композитний відеовихід, поєднаний з 3.5мм аудіовиходом (для підключення RCA екрану необхідно використовувати спеціальний кабель)				HDMI **
Audi Out.	3.5 mm джек, HDMI						Багатоканальний HD звук через HDMI

1	2	3	4	5	6	7	8
Audi In.	Немає вбудованого аудіовходу, але підтримуються USB звукові карти з аудіовходом						
Карта пам'яті	SD / MMC / SDIO		MicroSD				
Мережа	-	10/100 Ethernet RJ45	-	10/100 Ethernet RJ45		10/100 Ethernet RJ45, встроений 802.11n Wi-Fi и Bluetooth 4.1	-
Низькорівнева периферія	26 вивідний контактний GPIO роз'єм, що включає в себе загальні порти введення / виводу, а також інтерфейси SPI, I²C, I²S, UART		40 вивідний контактний GPIO роз'єм, що включає в себе загальні порти введення / виводу, а також інтерфейси SPI, I²C, I²S, UART				40 вивідний контактний GPIO роз'єм, що включає в себе загальні порти введення / виводу, а також інтерфейси SPI, I²C, I²S, UART. ***
RTC	Вбудовані RTC відсутня ****						
Енергоспоживання	300 мА, (1.5 Вт)	700 мА, (3.5 Вт)	600мА - 1.2А	~650мА, (3.0 Вт)		800мА (4.0 Вт) - 2.5А	160мА
Живлення	5V (DC) через роз'єм MicroUSB або GPIO						
Габарити	65x56x15мм	85x56x17мм	65x56x12мм	85x56x17мм	85x56x17мм	85.6x56.5x17мм	65.0x30.0x5мм
Вага	31 г	40 г	23 г	40 г	40 г	45 г	9 г

* - 256 МВ (до 15.10.2012) і 512МВ (з 15.10.2012)

** - композитний відеовихід не припаяний, передбачені 2 отвори для його монтажу

*** - Роз'єм не припаяний, потрібно самостійний монтаж

**** - Можна підключити RTC модуль по I²C

Розташування виводів SB LattePanda 2Гб



ДОДАТОК Д

Порівняльна таблиця ODROID-C2 з аналогічними мінікомп'ютерами

	ODROID-C2	ODROID-C1+	RPi 2 Model B
CPU	Amlogic S905 SoC 4 x ARM Cortex-A53 2GHz 64bit ARMv8 Architecture @28nm	Amlogic S805 SoC 4 x ARM Cortex-A5 1.5GHz 32bit ARMv7 Architecture @28nm	Broadcom BCM2836 4 x ARM Cortex-A7 900MHz 32bit ARMv7 Architecture @40nm
GPU	3 x ARM Mali-450 MP 700MHz	2 x ARM Mali-450 MP 600MHz	1 x VideoCore IV 250MHz
RAM	2GB 32bit DDR3 912MHz	1GB 32bit DDR3 792MHz	1GB 32bit LP-DDR2 400MHz
Flash Storage	Micro-SD UHS-1 @83Mhz/SDR50 or eMMC5.0 storage option	Micro-SD UHS-1 @78Mhz/SDR50 or eMMC4.5 storage option	Micro-SD @ 50Mhz/SDR25 No eMMC storage option
USB2.0 Host	4 Ports		
USB2.0 Device /OTG	1 Port for Linux USB Gadget device or USB host		No
Ethernet / LAN	10 / 100 / 1000 Mbit/s		10 / 100 Mbit/s
Video Output	HDMI 2.0 4K / 60Hz	HDMI 1.4	HDMI 1.4 / RCA / DSI
Audio Output	HDMI / I2S		MDMI / 3.5mm Jack / I2S
Camera Input	USB 720p		MIPI CSI 1080p
Real Time Clock	No (unless using an add-on module)	Yes (on-board RTC)	No (unless using an add-on module)
IR Receiver	Yes (on-board IR sensor)	Yes (on-board IR sensor)	No (unless using an add-on module)
IO Expansion	40 + 7 pin port GPIO / UART / I2C / I2S / ADC	40 + 7 pin port GPIO / UART / SPI / I2C / I2S / ADC	40 pin port GPIO / UART / SPI / I2S
ADC	10 bit SAR 2 channels		No (unless using an add-on board)
Heat sink	Included		Optional
Size	85 x 56 mm (3.35 x 2.2 inch)	85 x 56 mm (3.35 x 2.2 inch)	85 x 56 mm (3.35 x 2.2 inch)
Weight	40g	40g	42g

ДОДАТОК Е

Програма сервісної частини дистанційного керування мобільним роботом

```
import RPi.GPIO as GPIO
import socket
import time

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(33, GPIO.OUT)
GPIO.setup(11, GPIO.OUT)
GPIO.setup(13, GPIO.OUT)
GPIO.setup(15, GPIO.OUT)
GPIO.setup(29, GPIO.OUT)
GPIO.setup(31, GPIO.OUT)
GPIO.output(29, True)
GPIO.output(31, True)

UDP_IP = "0.0.0.0"
UDP_PORT = 5050

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.bind((UDP_IP, UDP_PORT))

while True:
    raw=sock.recvfrom(1024)

    #print type(raw)
    rawnew = ('forward', 'stop', 'backward', 'left', 'right' )

    print type(rawnew)

    if raw[0] == rawnew[0]:
        print 'robot moves forward'
        GPIO.output(33, True)
        GPIO.output(11, False)
        GPIO.output(13, True)
        GPIO.output(15, False)
        time.sleep(1)

    elif raw[0] == rawnew[1]:
        print 'robot stop'
        GPIO.output(33, False)
```

```

GPIO.output(11, False)
GPIO.output(13, False)
GPIO.output(15, False)
time.sleep(1)

elif raw[0] == rawnew[2]:
    print "robot moves backward"
    GPIO.output(33, False)
    GPIO.output(11, True)
    GPIO.output(13, False)
    GPIO.output(15, True)
    time.sleep(1)

elif raw[0] == rawnew[3]:
    print "robot moves left"
    GPIO.output(33, False)
    GPIO.output(11, True)
    GPIO.output(13, True)
    GPIO.output(15, False)
    time.sleep(1)

elif raw[0] == rawnew[4]:
    print "robot moves right"

    GPIO.output(33, True)
    GPIO.output(11, False)
    GPIO.output(13, False)
    GPIO.output(15, True)
    time.sleep(1)
else:
    print "message error"
    GPIO.output(33, False)
    GPIO.output(11, False)
    GPIO.output(13, False)
    GPIO.output(15, False)
    time.sleep(1)

```

ДОДАТОК Ж

Програмний засіб серверної частини системи дистанційного керування
мобільного робота

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

using RoboControl.Presenter;

namespace RoboControl
{
    public partial class Form1 : Form
    {
        Presenter.Presenter presenter;

        public Form1()
        {
            InitializeComponent();
        }

        public void ShowMessage(string text)
        {
            if (text != null)
            {
                MessageBox.Show(text);
            }
        }

        private void bn_appy_configs_Click(object sender, EventArgs e)
        {
            if (tb_port.Text.Length == 4 && tb_host.Text.Length > 5)
            {
                presenter = new Presenter.Presenter(this, Convert.ToInt32(tb_port.Text),
                tb_host.Text);

                var page = @"

```

```

        <iframe src='http://'+ tb_host.Text + "' width='100%' height='100%'"> </
        iframe > ";
        webBrowser.DocumentText = page;
        webBrowser.ScriptErrorsSuppressed = true;

        MessageBox.Show("Configs has been saved");
    }
    else
    {
        MessageBox.Show("Uncorrect value");
    }
}

private void bn_event(object sender, EventArgs e)
{
    switch( ((Button)sender).Name )
    {
        case "bn_command_1":
            presenter.Send_command("value", Commands.FORWARD);
            break;
            case "bn_command_2":
                presenter.Send_command("value", Commands.BACKWARD);
                break;
            case "bn_command_3":
                presenter.Send_command("value", Commands.LEFT);
                break;
            case "bn_command_4":
                presenter.Send_command("value", Commands.RIGHT);
                break;
            case "bn_command_5":
                presenter.Send_command("value", Commands.STOP);
                break;
            default:
                break;
    }
}

private void Form1_FormClosed(object sender, FormClosedEventArgs e)
{
    MessageBox.Show("connection has been closed");
    if (presenter != null)
    {
        presenter.CloseConnection();
    }
}

```

```

private void bn_KeyDown(object sender, KeyEventArgs e)
{
    switch (((Button)sender).Name)
    {
        case "bn_command_1":
            presenter.Send_command("value", Commands.FORWARD);
            break;
        case "bn_command_2":
            presenter.Send_command("value", Commands.BACKWARD);
            break;
        case "bn_command_3":
            presenter.Send_command("value", Commands.LEFT);
            break;
        case "bn_command_4":
            presenter.Send_command("value", Commands.RIGHT);
            break;
        case "bn_command_5":
            presenter.Send_command("value", Commands.STOP);
            break;
        default:
            break;
    }
}

```

```

private void bn_KeyPress(object sender, KeyPressEventArgs e)
{
    switch (((Button)sender).Name)
    {
        case "bn_command_1":
            presenter.Send_command("value", Commands.FORWARD);
            break;
        case "bn_command_2":
            presenter.Send_command("value", Commands.BACKWARD);
            break;
        case "bn_command_3":
            presenter.Send_command("value", Commands.LEFT);
            break;
        case "bn_command_4":
            presenter.Send_command("value", Commands.RIGHT);
            break;
        case "bn_command_5":
            presenter.Send_command("value", Commands.STOP);
            break;
        default:
            break;
    }
}

```

```

}
}
}
class Presenter
{

    Form1 form;
    int socet_port;
    Socket socet;

    public Presenter(Form1 form, int port, string host)
    {
        this.form = form;
        this.socet_port = port;

        socet = new Socket(AddressFamily.InterNetwork, SocketType.Dgram,
            ProtocolType.Udp);

        try
        {
            socet.Connect(host, port);
            this.form.ShowMessage("Connecting");
        }
        catch(Exception ex)
        {
            this.form.ShowMessage(ex.ToString());
        }
    }

    public Bitmap GetFrame()
    {
        var width = 410;
        var height = 325;

        Bitmap bmp = new Bitmap(width, height);

        if (socet != null)
        {
            using (Graphics gfx = Graphics.FromImage(bmp))
            using (SolidBrush brush = new SolidBrush(Color.FromArgb(0, 255, 0)))
            {
                gfx.FillRectangle(brush, 0, 0, width, height);
            }
        }
    }
}

```

```

return bmp;
}

public void Send_command(string value, Commands command)
{
    if (socet != null)
    {
        byte[] send_buf = new byte[200];

        switch (command)
        {
        case Commands.FORWARD:
            send_buf = Encoding.ASCII.GetBytes("forward");
            break;
        case Commands.BACKWARD:
            send_buf = Encoding.ASCII.GetBytes("backward");
            break;
        case Commands.LEFT:
            send_buf = Encoding.ASCII.GetBytes("left");
            break;
        case Commands.RIGHT:
            send_buf = Encoding.ASCII.GetBytes("right");
            break;
        case Commands.STOP:
            send_buf = Encoding.ASCII.GetBytes("stop");
            break;
        default:
            break;
        }
        socet.Send(send_buf);
    }
}

public void CloseConnection()
{
    if (socet != null)
    {
        socet.Close();
    }
}

enum Commands
{
    FORWARD, BACKWARD, LEFT, RIGHT, STOP
}
}

```


ДОДАТОК И

Скрипт Python для ультразвукового далекоміра

```

import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
TRIG = 16
ECHO = 18
GPIO.setup(TRIG, GPIO.OUT, initial=0)
GPIO.setup(ECHO, GPIO.IN)

# Minimum delay between measurements is 50ms. 100ms is fine:
time.sleep(0.1)

# Send a probe signal to the sensor. Signal should be 10 micro sec long.
GPIO.output(TRIG,1)
time.sleep(0.00001)
GPIO.output(TRIG,0)

while GPIO.input(ECHO) == 0:
    pass
start = time.time()

while GPIO.input(ECHO) == 1:
    pass stop = time.time()

# Print distance to object in santimeters. Sound speed = 340 m/s
print "Distance = ",(stop - start) * 17000,"sm"
print "start time = ", start
print "stop time = ", stop

GPIO.cleanup()

-----

$ cat 5_ultrasonic_distance_02_infinet-loop.py
import RPi.GPIO as GPIO
import time #GPIO.cleanup()

GPIO.setmode(GPIO.BOARD)
TRIG = 16
ECHO = 18

```

```

GPIO.setup(TRIG, GPIO.OUT, initial=0)
GPIO.setup(ECHO, GPIO.IN)

try:
    while True:
        # Minimum delay between measurements is 50ms. 100ms is fine:
        time.sleep(0.1)

        # Send a probe signal to the sensor. Signal should be 10 micro sec long.
        GPIO.output(TRIG,1)
        time.sleep(0.00001)
        GPIO.output(TRIG,0)

        while GPIO.input(ECHO) == 0:
            pass
            start = time.time()

        while GPIO.input(ECHO) == 1:
            pass
            stop = time.time()

        # Print distance to object in santimeters. Sound speed = 340 m/s
        print "Distance = ",(stop - start) * 17000,"sm"
        print "start time = ", start
        print "stop time = ", stop

except KeyboardInterrupt:
    GPIO.cleanup()

```

ДОДАТОК К

Скрипт Python для одночасного управління двома сервомоторами

```

from time import sleep
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

pan = 27
tilt = 17

GPIO.setup(tilt, GPIO.OUT) # white ==> TILT
GPIO.setup(pan, GPIO.OUT) # gray ==> PAN

def setServoAngle(servo, angle):
    assert angle >= 30 and angle <= 150
    pwm = GPIO.PWM(servo, 50)
    pwm.start(8)
    dutyCycle = angle / 18. + 3.
    pwm.ChangeDutyCycle(dutyCycle)
    sleep(0.3)
    pwm.stop()

if __name__ == '__main__':
    import sys
    if len(sys.argv) == 1:
        setServoAngle(pan, 90)
        setServoAngle(tilt, 90)
    else:
        setServoAngle(pan, int(sys.argv[1])) # 30 ==> 90 (middle point)
        ==> 150
        setServoAngle(tilt, int(sys.argv[2])) # 30 ==> 90 (middle point) ==>
        150
    GPIO.cleanup()

```

Коли скрипт виконується, Ви повинні ввести параметри, кут повороту і кут нахилу. наприклад:

```
sudo python3 servoCtrl.py 45 120
```

Вище зазначена команда буде позиціонувати механізм повороту/нахилу з 45 градусами за «азимуту» (кут повороту) і 120 градусів "висота" (кут нахилу). Зверніть увагу, що, якщо параметри не було введено, за замовчуванням буде встановлено обидва кута повороту і нахилу до 90 градусів.