Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Другий (магістерський)
(рівень вищої освіти)

Розроблення програмного забезпечення маніпуляційної системи мобільного робота Festo Robotino
(тема)

Виконав:
студент 2 курсу, групи АУТПм-20-2

Усенко Р. К.
(прізвище, ініціали)

Спеціальності 151 Автоматизація та комп`ютерно-інтегровані технології
(код і повна назва спеціальності)

Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Автоматизоване управління технологічними процесами
(повна назва освітньої програми)

Керівник проф. Цимбал О.М.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри КІТАМ _____ Невлюдов І. Ш.
(підпис) (прізвище, ініціали)

2021 р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАМ
Рівень вищої освіти другий (магістерський)
Спеціальність 151 Автоматизація та комп`ютерно-інтегровані технології
Тип програми Освітньо-професійна
Освітня програма Автоматизоване управління технологічними процесами
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАМ _____
(підпис)
«_____» _____ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Усенку Руслану Костянтиновичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення програмного забезпечення маніпуляційної системи мобільного робота Festo Robotino
Затверджена наказом по університету від 08.11.2021 р. №1699 Ст.
2. Термін подання студентом роботи до екзаменаційної комісії 12.12.2021 р.
3. Вихідні дані до роботи мобільна платформа Festo Robotino, система програмування Robotino View, система моделювання Robotino SIM, система розробки Visual Studio 2017
4. Перелік питань, що потрібно опрацювати в роботі
4.1 Аналіз технічного завдання;
4.2 Аналіз робототехнічних систем
4.3 Аналіз симуляторів робототехнічних систем
4.3. Розгляд систем технічного зору
4.4 Аналіз програмного забезпечення Robotino View
4.5 Розробка алгоритмів та програм
4.6 Аналіз отриманих результатів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів). Демонстраційний матеріал, представлений у форматі презентації PowerPoint (*.ppt) – 12 сторінок формату А4

6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз технічного завдання	15.06.21	виконав
2	Аналіз робототехнічних систем	19.09.21	виконав
3	Аналіз симуляторів робототехнічних систем	24.09.21	виконав
4	Розгляд систем технічного зору	15.10.21	виконав
5	Аналіз програмного забезпечення Robotino View	12.11.21	виконав
6	Оформлення пояснювальної записки	19.11.21	виконав
7	Подання роботи на перевірку Інтернет-сервісом Unicheck	18.12.21	виконав
8	Подання роботи на рецензію	18.12.21	виконав
9	Подання роботи на підпис зав. кафедри	.21	виконав
10	Подання роботи до ЕК	.21	виконав

Дата видачі завдання 02.09.2021

Студент _____ Усенко Р.К.
(підпис) (прізвище, ініціали)

Керівник роботи_____ проф. Цимбал О.М.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 86 с., 2 табл., 53 рис., 1 дод., 27 джерел.

МОБІЛЬНИЙ РОБОТ, МОБІЛЬНІ ПЛАТФОРМИ, НАВІГАЦІЯ.

Об'єкт дослідження – система керування мобільного робота Festo Robotino.

Предмет дослідження – програмне забезпечення маніпуляційної системи мобільного робота Festo Robotino.

Мета магістерської роботи – розробка удосконаленого програмного забезпечення для маніпуляційної системи мобільного робота

В роботі проведено аналіз алгоритмів уникнення перешкод під час руху Розглянуті підходи, що можна використати для корегування траєкторії руху.

Розглянуті можливості використання компьютерного зору

За результатами аналізу було розроблено алгоритм переміщення по заданим координатам, рух по лініям чорного кольору та пошук об’єктів певного кольору,

ABSTRACT

Explanatory note: 86 pages, 2 table, 53 figures, 1 app, 27 sources.

MOBILE ROBOT, MOBILE PLATFORMS, NAVIGATION.

The object of research is the Festo Robotino mobile robot control system.

The subject of the research is the software of the Festo Robotino mobile robot manipulation system.

The purpose of the master's thesis is to develop advanced software for the manipulation system of the mobile robot

The analysis of algorithms to avoid obstacles during movement is carried out in the work. Approaches that can be used to adjust the trajectory are considered.

Possibilities of using working vision are considered

Based on the results of the analysis, an algorithm for moving along the given coordinates, moving along black lines and searching for objects of a certain color was developed.

The object of research - the process of movement of coordinates for the mobile robot Festo Robotino.

The subject of research is the software module of the mobile robot Festo Robotino.

The purpose of the master's thesis is to search for and move objects using the Festo mobile transport robot.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ЕОМ – електронна обчислювальна машина;

МР – мобільний робот;

ОС – операційна система;

ПЗ – програмне забезпечення;

ПК – персональний комп’ютер;

СКЗ – система комп’ютерного зору;

API – application programming interface;

GUI – Graphical User Interface;

SDK – Software Development Kit.

ВСТУП

На даний час існує велика кількість інфекційних хвороб и стоіть актуальне питання захисту медицинських працівників що постійно контактують з хворими від випадкового зараження Для вирішення цього питання, використовують різноманітні підходи, одним з яких є використання повністю автономних роботів.

У цій роботі представлено мобільного робота, на платформі Festo Robotino з інтегрованим маніпулятором який виконує логістичні функції а саме доставку фармацефтичних препаратів без втручання людини що в свою чергу дасть змогу минімізувати контакт з хворими. Основним інструментом розробки виступає симулятор робототехнічних систем Robotino View та Robotino SIM. Таким чином напрямок даної роботи є актуальним.

Об’єкт дослідження – система керування мобільного робота Festo Robotino.

Предмет дослідження – розробка програмного забезпечення маніпуляційної системи на базі мобільного робота Festo Robotino.

Метою випускної магістерської роботи є розробка програмного забезпечення для мобільного транспортного робота Festo Robotino у середовищі розробки Robotino View.

Досягнення мети полягає у виборі оптимального методу розпізнавання обектів по кольору та транспортування їх в кінцеву точку

Для реалізації поставленої мети необхідно вирішити наступні завдання:

– провести аналіз використання мобільних роботів;

– провести аналіз використання мобільних роботів;

– провести аналіз сучасних симуляторів мобільних роботів;

– розглянути програмні засоби мобільних роботів;

– дослідити методи руху та технічного зору мобільного робота;

– розробити програмний модуль;

– протестувати компоненти програмного модуля.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Мобільні роботи

Мобільний робот - це автоматичний механізм, здатний рухатися в навколишньому просторі. Така інтелектуальна технічна система, яка, виконує вказані дії відповідно до бази знань, інтегровану в ній. Залежно від програму-програмного коду, закладеного в блоці керування, мобільний робот діє автономно або контролюється на відстані оператором. Механізм рухається вздовж даного алгоритму або незалежно визначає шлях руху.

Типізація робототехніка побудована за параметрами автономних пристроїв та їх здібностей взаємодіяти з навколишнім середовищем.

Мобільні роботи можна класифікувати середовищем використання.

– наземні: колісні, безпілотні автомобілі, транспортні роботи-навантажувачі;

– повітряні, включно з компактними дронами та вертольотами високої потужності, керовані автопілотом;

– морські – Superwater - човни з автономним або радіо-контролем.

1.2. Застосування сучасних мобільних роботів у фармацевтиці

Сьогодні інновації – це невід'ємна частина нашого життя. Це те, що рухає прогрес. Якщо ще 20-30 років тому ми стикалися з інноваціями раз на кілька років, то зараз практично по кілька разів на день. Вони допомагають оптимізувати будь-які процеси, тим самим полегшуючи наше життя. Крім цього, інноваційна діяльність приносить користь державі тим, що сприяє збільшенню ВВП. Зараз велику популярність мають наскрізні цифрові технології, які безпосередньо пов'язані з інноваціями. До них відносяться: великі дані, віртуальна реальність, інтернет речей, блокчейн, бездротовий зв'язок, робототехніка. На рис. 1.1 – основні типи цифрових технологій [1].

Виходячи з даних на графіку одним з головних лідерів є роботизація. Робототехніка - одна з наук, що стрімко розвиваються, яку можна віднести до проривної технологічної інновації, що володіє значними перспективами в розвитку. Вона займається проектуванням автоматичних технологічних систем і є дуже важливою технічною основою сучасного виробництва.

Рисунок 1.1 – Основні типи цифрових технологій

Обсяг ринку роботів до 2020 року оцінюється в 500 млрд. доларів. Щорічно з'являється тисячі роботів, які значно спрощують життя людині, і вже стає складно знайти сферу діяльності, в якій тією чи іншою мірою не застосовується робототехніка. Виділяють будівельну, промислову, побутову, медичну, авіаційну та екстремальну (військову, космічну, підводну) робототехніку.

Охорона здоров'я одна з прогресивних сфер, в якій застосовується роботи. В даний час активно розвивається роботизована хірургія, використовуються біонічні протези. Крім цього, робототехніка застосовується у фармацевтиці. Робот-фармацевт, він же аптечний робот - це складське обладна-

ння, яке нагадує торговий автомат. Даний робот призначається для продажу лікарських засобів.

Основне завдання даного робота - це прискорення процесу видачі ліків фармацевтом, або ж повністю його заміна, а також оптимізація зберігання. На рис. 1.2 – аптечний робот CONSIS Pack Dispensers.

Вперше робот був представлений ще в 1996 році на виставці в Мюнхені і призначався для автоматизації видачі найбільш затребуваних медикаментів в аптеці. Роботів фармацевтів виробляють німецькі компанії "Consis" та "Rowa", іспанська компанія "RePhar-ma", італійська компанія "Medistore", а також фінська компанія "Newico".

Рисунок 1.2 – Аптечний робот CONSIS Pack Dispensers

У результаті після впровадження робота-фармацевта скорочується час на логістичні операції з товаром від 4 год до 5 год на день на кожного фармацевта, знижується час обслуговування клієнтів, ліквідуються прострочені товари, фармацевт концентрується тільки на консультаційних питаннях.

Підбиваючи підсумки, можна дійти невтішного висновку, що використання робота в аптеку вимагає великих вкладень, але при цьому надає пози-

тивний ефект прискорюючи процес обслуговування, і дає величезну перевагу перед конкурентами.

1.3 Огляд сучасних симуляторів

На сьогоднішній день стрімко розвиваються засоби автоматизації різних процесів, все більш широко застосовуються засоби робототехніки. Згідно з прогнозом Японської асоціації робототехніки, через 5 - 10 років обсяг ринку персональної та домашньої робототехніки досягне десятків мільярдів доларів [14]. При цьому виходу робототехнічних систем на споживчий рівень перешкоджає низку проблем. Так, ринок робототехніки відокремлений, виробники розробляють різні пристрої окремо. Немає єдиного стандарту для застосовуваних роботизованих систем, внаслідок чого робототехнічні системи (або їх окремі підсистеми) є несумісними один з одним, а нові технології та інструмент, призначений для розробки робототехнічних систем, має вузьку специфікацію.

Крім іншого, область робототехніки досить складна і вимагає дуже високої кваліфікації, а обміну досвідом і знаннями між різними сферами робототехніки практично не відбувається [14]. Окремою проблемою є складність тестування та налагодження роботизованих систем. Ці процеси вимагають значних витрат часу та ресурсів, особливо при розробці роботів для застосування в спеціальних умовах, наприклад, підводних роботів або роботів для гасіння пожеж. Прискорити і здешевити проектування робототехнічної системи можна за допомогою попереднього тестування алгоритмів управління та конструювання віртуальної моделі такої системи. Засоби проектування, імітації та випробування робототехнічних систем узагальнюються у понятті «симулятор», що частково вирішує вищеописані проблеми.

Симулятори дозволяють вирішити проблему моделювання особливих (небезпечних або складно відтворюваних у реальних випробуваннях) умов, а також надають можливість одночасної роботи над проектом кільком членам

команди, що неможливо при роботі з реальною моделлю, оскільки вона зазвичай виконується в одиничному екземплярі.

Застосування симуляторів робототехніки в розробці робототехнічних систем є перспективним і активно розвивається в галузі програмних засобів підтримки автоматизації та роботизації виробництва.

1.3.1 Структура та класифікація симуляторів

Перш ніж перейти до аналізу існуючих фізичних та графічних інструментів моделювання елементів комп'ютерних робототехнічних систем, наведемо деякі основні поняття та визначення. Симулятор – імітатор, механічний або комп'ютерний, що відтворює керування будь-яким процесом, апаратом чи технічним засобом. У цьому випадку – робототехнічної системою. Також можливе інше трактування даного терміна: симулятори - набір програмних і апаратних засобів, що створюють враження дійсності, відображаючи частину реальних явищ і властивостей у віртуальному середовищі.

Симуляція - використання комп'ютерів для імітації реально існуючих ситуацій. Одні симулятори використовуються для навчання і тренування в управлінні машинами, інші є програмами, які пророкують (або роблять спробу передбачити) події реального світу [2]. Процес моделювання (симуляції) супроводжується відображенням реального об'єкта (системи) у модель, що змінює свій стан з часом. Причому час необоротний, він не сповільнюється і не прискорюється. Стан системи визначається станом її елементів, а кожен елемент має набір властивостей (характеристик). В імітаційному моделюванні термін «час» може означати різні поняття: а) фізичний (physical time, Tp) час – це час, який використовується у реальній (фізичній) системі, яку моделюють.

Модельне (simulation time, Ts) - це уявлення фізичного часу в моделі; за одиницю модельного часу (h) можна прийняти часовий інтервал в 1 хв,

10 хв, 30 хв, в одну годину і т.д. Ts = Tp/h. в) процесорний (wallclock time, Tw) – час роботи симулятора на комп'ютері. Однією з переваг систем моделювання є можливість задавати швидкість виконання моделювання. Вид моделювання, коли модельний час синхронізується із процесорним, називається моделюванням у реальному часі (real time) [8].

Одним із основних об'єктів симулятора вважається віртуальний робот. Віртуальний робот – це віртуальна модель, що імітує реальну модель роботехнічної системи, що відтворюється у віртуальному середовищі симулятора, де вона представлена у вигляді просторової моделі. Кожен симулятор включає фізичний та графічний двигун. Від їх можливостей залежить складність моделі робото або комп'ютера [8], яку можна реалізувати в симуляторі. Двигун (англ. engine – мотор, двигун) – центральна частина комп'ютерної програми, що виконує основні функції цієї програми. Залежно від контексту цього поняття може відповідати різний зміст. Графічний двигун - програма, основним завданням якої є візуалізація (рендеринг) двомірної або тривимірної комп'ютерної графіки. Графічний двигун працює в режимі реального часу і здійснює візуалізацію за допомогою графічних процесорів.

Фізичний движок (англ. physics engine) - комп'ютерна програма, яка виробляє комп'ютерне моделювання фізичних законів реального світу у віртуальному світі, з тим чи іншим ступенем апроксимації. Найчастіше фізичні двигуни для фізичного моделювання використовуються не як окремі самостійні програмні продукти, а як складові компоненти (підпрограми) інших програм. Фізичний двигун дозволяє створити віртуальний простір, в який можна додати віртуальні статичні та динамічні об'єкти та вказати закони взаємодії тіл та середовища. Розрахунок взаємодії тіл виконується самим двигуном. Обчислюючи взаємодію тіл між собою і з середовищем, фізичний двигун наближає фізичну модель одержуваної системи до реальної і передає уточнені геометричні дані графічному движку.

Симулятори можна умовно поділити на три групи:

– симулятори без графічного двигуна (текстові);

– симулятори, що мають графічний двигун (візуальні);

– симулятори, що мають графічний і фізичний двигун (універсальні).

Існує також спеціальне програмне забезпечення для роботи з роботами, яке є лише доповненням (інструментарієм) до симулятора. Це програмне забезпечення не оснащене ні фізичним, ні графічним двигунами. Існуючі різні симулятори мають як переваги, так і недоліки при їх використанні для комп'ютерного моделювання різних систем. Серед переваг слід зазначити:

– низьку вартість;

– можливість будь-якої миті доопрацювати модель;

– можливість окремо тестувати функціональні складові,

– можливість одночасної симуляції декількох типів роботів.

До недоліків можна віднести:

– у даний час навіть найдосконаліший фізичний двигун не може симулювати всі закони реального світу;

– вибагливість до ресурсів машини.

1.3.2 Огляд сучасних симуляторів

Розглянемо найбільш поширені симулятори, що застосовуються в робототехніці, можливості їх використання з метою управління різними роботами, а також для розробки та конструювання комп'ютерних роботооператорів. Слід згадати, що багато симуляторів підтримують лише певний вид роботів (певних виробників або мікропроцесорів). Охарактеризуємо найбільш відомі симулятори.

Пакет SolidWorks Simulation [21] - універсальний інструмент для аналізу міцності конструкцій методом кінцевих елементів. У ньому для вирішення задач симуляції забезпечується повноцінний статичний аналіз як деталі, так і складання з використанням кінцевих елементів твердого тіла, поверхонь та балок, реалізовані різноманітні контактні умови та всілякі віртуальні з'єднувачі.

Ця система є почасти візуальним симулятором. Вона в основному використовується для проектування конструкції робота. Підтримує інтеграцію та сумісність із багатьма популярними симуляторами; підтримує інтеграцію в інші симулятори шляхом експорту даних у файли з розширенням *.SLDASM, *.SLDPR, *.xml, *.stl, *.SLDPRT, *sdf, *wrl, *.dae. Відсутня підтримка мов програмування, тобто відсутня інструментарій управління роботом. Є платним товаром.

NXT-G [6] є графічним симулятором. Створено спеціально для популярного робототехнічного комплекту Lego Mindstorms NXT. Використовується для програмування NXT Brick.

Цей симулятор має інтуїтивно зрозумілий інтерфейс, управління роботами здійснюється за допомогою спеціальних блоків, що розміщуються на LEGO-балках вздовж осі послідовності дій. Порядок виконання програми визначається порядком прямування блоків. Є безкоштовним продуктом.

TrikStudio [7] – графічний симулятор, орієнтований програмування навчальних роботів фірми Trik. Включає в себе симулятор, який дозволяє тестувати програми без використання реального робототехнічного набору. Відмінною особливістю TRIK Studio є інтерактивний режим імітаційного моделювання. Є безкоштовним продуктом.

Найбільшим і відомим симулятором є комбінація пакета Simulink та Robotics System Toolbox [8], що входять до складу програмного продукту MATLAB. Даний продукт забезпечує алгоритми та апаратні підключення для розробки автономних мобільних робототехнічних додатків. Цей пакет має велику бібліотеку різноманітних технічних засобів і пристроїв, більшість з яких доступна для випробування на персональному комп'ютері. Алгоритми Toolbox включають представлення карти руху, планування маршруту, і подальшого шляху для мобільних роботів. Також можна проектувати і здавати прототипи управління двигуном, реалізовувати комп'ютерний зір для робота. Підтримує як інтеграцію сторонніх проектів симуляторів і CAD-систем, а саме форматів *.xml, *.stl, а також різні GUI-інтерфейси ; підтримує

мову програмування C/C++. Належить до категорії універсальних симуляторів. Є платним товаром.

AnyKode Marilou Robotics Studio (далі – Marilou) [9] – середовище розробки та симулювання мобільних роботів та маніпуляторів з урахуванням фізичних законів реального світу. Підтримує контроль та управління над наступними фізичними параметрами: маса, пружність, розміри матеріалу, крутний момент і т.п. Marilou дозволяє підключати до роботи різні віртуальні пристрої: компас, акселерометри, двигуни та сервомотори, бампер, сенсори відстані (ультразвуковий та інфрачервоний), GPS та інші пристрої. У редакторі об'єктів Marilou доступні статичні та динамічні об'єкти, які можна розміщувати у світі, що симулюється (підтримується одночасна симуляція кількох роботів). Складні об'єкти в Marilou будуються з більш простих (використовується ієрархічний підхід до представлення об'єкта), що дозволяє повторно використовувати частини об'єктів. Основою Marilou є технологія MODA (Marilou Open Devices Access) – SDK для роботи з роботами та їх компонентами у симуляторі. Після синхронізації із симулятором алгоритми керування роботом можуть запускатися на комп'ютері. Програмування алгоритмів управління роботів можливе з допомогою мов C/C++, C++ CLI, C#, J#, VB#. Є умовно-безкоштовним товаром.

Arduino IDE [10] є текстовим симулятором, який дозволяє складати програми у зручному текстовому редакторі, компілювати їх у машинний код, і завантажувати на різні версії мікропроцесорів Arduino. Є без-платним та кросплатформним продуктом, підтримує популярні мови програмування: C/C++, C#, Java, Python.

OROCOS (Open Robot Control Software project) [11] – текстовий симулятор. Є безкоштовним продуктом і використовується як додатковий інструмент для різних симуляторів. Підтримує мову програмування C/C++. ROS.

(Robot Operating System) [12] - є універсальним симулятором з відкритим вихідним кодом (ліцензія BSD - тобто можливо використовувати і модифікувати програму під свої завдання, в т.ч. в комерційних цілях).

Головна можливість ROS – це повторне використання коду у робототехнічних дослідженнях та розробках. Цей продукт підтримує інтеграцію різних драйверів, алгоритмів і популярних відкритих робототехнічних бібліотек. Крім цього ROS володіє великим функціоналом: апаратна абстракція, низькорівневий контроль обладнання, реалізація часто використовується функціоналу, передача повідомлень між процесами, управління пакетами.

ROS не є системою реального часу, хоча може використовувати системи реального часу (наприклад, OROCOS Real-time Toolkit). ROS є розподіленою системою процесів. Також даний продукт легко інтегрується з різними симуляторами (на даний момент інтегрований з OpenRAVE, OROCOS і Player). Підтримує мови програмування C++ і Python (також є тестові бібліотеки на LISP, Octave Java, Lua). Також ROS має вбудований пакет для тестування – rostest, що полегшує тестування додатків. В даний час ROS працює тільки під UNIX-подібними системами.

V-REP [13] є одним з кращих рішень серед симуляторів, має велику кількість функцій, а також складний API. Універсальний симулятор робототехніки V-REP, з інтегрованим середовищем розробки, заснований на розподіленій архітектурі управління: кожен об'єкт/модель може керуватися індивідуально за допомогою вбудованого сценарію, плагіна, вузла ROS, клієнта віддаленого API або користувача рішення. Підходить для застосування у кількох роботах. Програмний код може бути написаний на C/C++, Python, Java, Lua, Matlab, Octave, але генерацію коду потрібно реалізувати через API, що є трудомістким процесом. Використовується для швидкої розробки алгоритмів, моделювання автоматизації виробництва, швидкого прототипування і верифікації робототехніки, віддаленого моніторингу і т.д. Підтримує інтегрування сторонніх симуляторів, а також різні GUI-інтерфейси та користувальницькі драйвера.

RobotC [14] – текстовий симулятор, є лідером серед мов програмування для вивчення роботів та підготовки до змагань. Він заснований мовою програмування З, і має простий у використанні середовищем розробки. Є платним програмним забезпеченням.

BricxCC [15] – текстовий симулятор, найпоширеніший інструмент, що підтримує мову програмування NXC. Це програма, що вільно розповсюджується, що має велику кількість різних інструментів для роботи з блоками Lego Mindstorms, фактично може повністю замінити стандартне програмне забезпечення Lego (крім драйверів). Вбудовані бібліотеки мови дозволяють працювати з пристроєм на різних рівнях, присутні низькорівневі засоби звернення до входів і виходів пристрою, звернення до фізичних адрес пам'яті, а також високорівневі команди управління моторами та отримання даних з датчиків.

Robotino View [16] - універсальний симулятор, створений Festo Didactic для навчання робототехніки. Програмувати робота можна з допомогою сучасних мов програмування: C/C++, C#, Java, і навіть є можливість інтеграції симулятора з MATLAB, ROS, SmartSoft, MRDS. Є безкоштовним продуктом.

Microsoft Robotics [17] - це пакет програм, який може використовуватися для управління різними роботами і включає повноцінний універсальний симулятор. Підтримує мови програмування VPL, C#. Цей симулятор дозволяє працювати з наступними віртуальними пристроями: GPS, лазерний далекомір, інфрачервоний далекомір, компас, сенсор кольору, сенсор яскравості, веб-камера. Є умовно-безкоштовним товаром. Компоненти в Robotics Studio представлені як незалежно виконуваних сервісів як сервісів з GUI-интерфейсом. Протокол SOAP, що використовується для взаємодії розподілених сервісів, не призначений для додатків, що працюють у режимі реального часу. Підтримує імпорт таких типів CAD-файлів: *.dae, *.obj і *.x. Крім цього підтримує інтеграцію в різні симулятори через GUI-і COM-інтерфейси та користувальницькі драйвера.

Gazebo [18] - Універсальний симулятор, розроблений для операційної системи Linux. Є безкоштовним продуктом. Дане рішення підтримує симуляцію кількох роботів з сенсорами в оточенні різних об'єктів, а також підтримує наступні моделювані сенсори: лазерний далекомір, камера, пристрій для читання RFID-міток. Підтримується більшість робототехнічних систем. У Gazebo вбудована можливість читання файлів у форматі Collada, що дозволяє додавати до симулятора об'єкти, спроектовані в одному з редакторів 3D-моделей. Підтримує мови програмування C/C++, Python. Підтримує як експорт власних проектів у сторонні симулятори, так і імпорт сторонніх симуляторів за допомогою форматів *.xml, *.stl, а також через GUI-інтерфейси та драйвери користувача.

Algodoo [19] – універсальний 2D-симулятор. У цьому продукті можна моделювати різні фізичні об'єкти, пружини, оптичні пристрої тощо. В Algodoo вбудовано скриптову мову програмування Thyme. Є безплатним продуктом.

URBI [20] це текстовий симулятор. Є кросплатформним рішенням, підтримує мову програмування C/C++. URBI ґрунтується на розподіленій компонентній архітектурі UObject. Вона також включає urbiScript - паралельний і подієвий скриптовий мову. В основі URBI лежить метод, що відокремлює програму, що управляє, на urbiScript від взаємодії з ОС і т.д. за допомогою прошарку з UObject-драйверів. Тобто. Для контролю робота за допомогою URBI потрібно створити проміжні UObject-драйвери для свого обладнання. Має інтеграцію з різними симуляторами.

USARSim (Unified System for Automation and Robotics Simulation) [21] є універсальним симулятором і безкоштовним рішенням, заснований на технології Unreal Engine 2.0, але без придбаної копії Unreal Engine, яка платна, не функціонує. Контроль над роботами, змодельованими в USARSim, може бути здійснений будь-якою мовою програмування, яка підтримує TCP сокети. Є кросплатформним рішенням. Підтримуються різні геометричні моделі та об'єкти для створення моделі робота. Крім цього підтримує моделювання рі-

зних інтерфейсів (GPS і т.п.). USARSim застосовується при моделюванні роботів, що використовуються для збору даних з Марса та Місяця.

Webots [22] є універсальним симулятором. Цей продукт використовується фізичний двигун Open Dynamics Engine (ODE). Є платним продуктом. Передбачає моделювання кількох роботів в одній сцені запуску. Підтримує програми C/C++, Java, Python. Webots дозволяє моделювати робота за допомогою різних геометричних моделей і об'єктів, а також за допомогою різних інтерфейсів і датчиків. Має бібліотеку створених моделей різних роботів.

RobotSim [23] є універсальним симулятором, який дозволяє моделювати робота, використовуючи різні об'єкти і датчики. Крім цього, це рішення передбачає імпорт CAD-файлів (COLLADA або 3DS). Управління моделлю робота забезпечується C++ API, LabVIEW, або іншими мовами програмування, що підтримують роботу з сокетами. Є платним рішенням.

SARGE (Search and Rescue Game Engine) [24] є універсальним симулятором. Він був розроблений для навчання правоохоронних органів у використанні робототехніки в пошуково-рятувальних роботах. Має ліцензію Apache License V2.0. Як графічний движок використовується Unity, а як фізичний движок - PhysX. SARGE підтримується операційними системами Windows та Mac. Крім того, при моделюванні робота є можливість роботи з такими пристроями, як 3D-камера, компас, GPS і т.п. Але, наприклад, для використання GPS-систем потрібна підтримка Google Earth.

lpzrobots [25] є універсальним симулятором, який знаходиться під GPL ліцензією (безкоштовний продукт з відкритим вихідним кодом), а також розповсюджується для платформ Linux та Mac. Цей продукт використовує ODE і OSG (OpenScreenGraph) двигуни.

SimRobot [26] є безкоштовним універсальним симулятором. Використовує ODE та OpenGL двигуни. Має библиотекою різних об'єктів робототехніки, але крім цього має інструментарій для створення користувальницьких роботів, а також містить обмежений набір різних пристроїв і датчиків:

камера, датчик відстані, датчик тиску та датчик, що визначає значення кутів повороту моторів.

SubSim [27] є текстовим симулятором, що застосовується при використанні контролерів EyeBot в автономних підводних апаратах. Використовує фізичний двигун Physics Abstraction Layer (PAL). Моделі різних апаратів можна імпортувати з CAD-файлів Milkshape3D. Для програмування використовується мова C/C++.

EyeSim [28] є двовимірним текстовим симулятором, що застосовується для робочих платформ EyeBot. Для реалізації функцій симулятора використовується бібліотека RoBIOS (Robot BIOS). Для створення моделі робота і навколишнього середовища використовує текстові файли Wall, що використовує чотири значення для представлення початкової та кінцевої позиції в двовимірній системі координат), або Maze, в якому буквально малюється карта руху в текстовому файлі за допомогою символів підкреслення, а також інші символи . Порівняємо характеристики симуляторів окремо розглядаючи симулятори різних типів: текстові, графічні, універсальні. . Всі вони характеризуються низькою ресурсомісткістю, відносно простим і інтуїтивно зрозумілим інтерфейсом. Більшість їх використовує як апарат управління мови програмування C/C++, практично відсутня можливість експорту/імпорту проектів різних симуляторів; вузькоспрямованість. Ключовою властивістю графічних симуляторів є їхня інтерактивність; Крім цього, можна виділити такі характеристики: як і текстовим симуляторам, графічним властива вузькоспрямованість; відсутня підтримка мов програмування. Універсальним симуляторам властива висока та середня ресурсомісткість; можливість роботи з будь-яким типом робота; наявність бібліотек з різними еталонами роботизованих механізмів; підтримка таких популярних мов програмування, як C/C++, C#, Python, Java; наявність інструментарію для тестування моделей роботів та різних інтерфейсів. Вони розділені за критерієм «ресурсоємність», яка визначається рівнем вимог до відеокарти. До групи симуляторів із середньою ресурсомісткістю віднесені системи з вимогами: 64 Мб VRAM,

1024 Мб RAM, 2,2 ГГц. До симуляторів з високою ресурсоємністю віднесені симулятори з вимогами: 256 Мб VRAM, 1024 Мб RAM, 2,2, ГГц. З проведеного аналізу можна назвати такі актуальні симуляторы: AnyKode Marilou Robotics Studio, V-REP, Gazebo.

Використання інструментарію симуляторів має ряд переваг: симулятори володіють графічним або фізичним двигуном, або їх комбінацією, які є важливим інструментарієм для здійснення симуляції, що надає розробникам можливість створювати або модифікувати вироблені робототехнічні системи без відриву від їх виробництва, дозволяє реалізувати оптимізувати різні ідеї до проведення тестів на реальну робототехнічну систему з метою їх впровадження; надає інструментарій для аналізу різних проектів (сценаріїв) та їх послідовностей; володіють інтуїтивно зрозумілим візуальним середовищем для демонстрації продукту, відображення реалізації ідеї розробки робототехнічної системи в різних середовищах і при використанні в різних процесах. Головним недоліком симуляторів є проблема з їхньою відповідністю реальним модельованим робототехнічним системам, а також навколишнім середовищам та різним процесам. Тобто це означає, що розробник повинен провести точне налаштування всіх параметрів в симуляторі на підставі реальної інформації.

Якщо вищезазначені дії будуть виконані, а також в ході симуляції віртуальний робот при взаємодії з середовищем і процесами, що моделюються, покаже позитивний результат, то код такої програми може бути впроваджений в реальному прототипі з мінімальними змінами.

1.4 Системи комп'ютерного зору

У робототехніці як і живому світі основним видом передавання відчуттів є зір.

Перші системи технічного (машинного, комп'ютерного) зору, що знайшли застосування в засобах робототехніки, копіювали органи зору живих

організмів, розвиваючись у такій послідовності: чорно-білі монокулярні СКЗ, кольорові, стереоскопічні та багаторакурсні з різними варіантами апаратної реалізації. стійке застосування в робототехніці СКЗ отримали в системах управління маніпуляторів і мобільних роботів людини-оператора.

Наступним кроком у розвитку таких систем стали системи супервізорного управління.

Останній випадок характерний, наприклад, для підводних роботів, що працюють поблизу дна, коли рух робота викликає підйом опадів і замутнення води.

Тому застосовується в цих випадках супервізорне управління зводиться до періодичного цілевказівки і завдання оператором чергової виконуваної автоматично елементарної операції в моменти відновлення видимості.

Тут вперше, принаймні вітчизняної практиці, було реалізовано групове застосування мобільних роботів з централізованим управлінням отоператора.

Оператор за допомогою рукояток, що задають, і різних засобів цілевказівки здійснював управління роботами-спостерігачами, які видавали на екран пульта управління загальну картину робочої зони, роботами-розвідниками, які проводили детальне обстеження цієї зони з передачею крім відео ще й іншої потрібної інформації, перш за все, звичайно, про радіаційну обстановку , і, нарешті, технологічними роботами, що виконували саму роботу з очищення приміщень і території станції від радіоактивного сміття. Подальшим етапом розвитку систем відчуття роботівстало комплексування СКЗ з іншими сенсорними системами,т. е. спільне їх використання перш за все для вирішення завдань, які інакше не могли бути вирішені.

Як приклад, на рис. 1.3 показана картина зони радіаційного зараження, одержувана на екрані пульта управління робота, призначеного для обстеження радіаційно заражених територій і пошуку радіоактивних джерел.

На екрані - картина ізоліній полярадіації, які побудовані за показаннями спеціальних сенсорів та за інформацією, що отримується від СКЗ.

Окрім отримання загальної картини цього поля і виявлених місць знаходження локальних джерел радіаційного випромінювання це дозволяє визначити величини активності останніх.

Рисунок 1.3 – Робочий оператор розвідки та схема зони забруднення випромінювання на пульті дистанційного керування

На рис. 1.4 показаний робот, призначений для боротьби. з терористами та виявлення різних небезпечних предметів. \

Він обладнаний рентгенівською системою просвічування.

На рисунку показано операцію перевірки вмісту портфеля та зображення, що отримується при цьому на пульті оператора.

Загальна тенденція розвитку СКЗ – це вдосконалення алгоритмів обробки відеоінформації, включаючи адаптацію до зовнішніх умов і вирішуваних завдань шляхом зміни структури та налаштування параметрів системи.

Актуальним завданням є перехід від бінокулярних СКЗ до багаторакурсних. У робототехніці це реалізується шляхом переміщення робота зі

СКЗ перпендикулярно напрямку бачення або такого ж переміщення відеокамери маніпулятором. Для окремих об'єктів — це огляд їх із усіх боків.

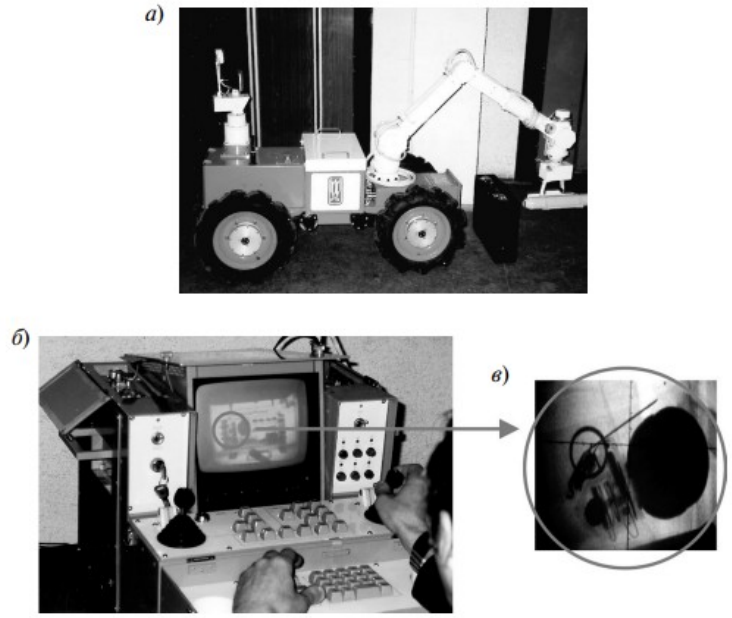

Рисунок 1.4 – Робот-сапер

В результаті комп'ютерної обробки одержуваної відеоінформації синтезується об'ємна модель. Зрозуміло, це незмірно інформативніше традиційних стереозображень і дозволяє здійснювати відповідно більш якісне планування та управління рухом. В даний час ведуться дослідження зі створення систем, які відтворювали б такий процес обробки вихідної відеоінформації аж до розпізнавання образів на основі асоціативного співвідношення з еталонними або конкретними образами або їх фрагментами.

СКЗ у складі систем управління роботів призначені поряд з іншими сенсорними системами насамперед для реалізації адаптивного управління та забезпечення людини-оператора, керуючого роботом, відеоінформацією про навколишній робот обстановці і насамперед про його робочу зону. У рамках вирішення останнього завдання СКЗ часто з´єднують з комп'ютерною графікою для побудови динамічної 3D картини зовнішнього середовища у вигляді системи стилізованої віртуальної реальності, яка використовується при пла-

нуванні дій робота, відпрацювання програм окремих операцій, а також при управління ними. Основні завдання, які вирішують СКЗ: отримання загальної зорової картини навколишньої зовнішнього середовища, виділення в цій картині окремих об'єктів та їх розпізнавання, включаючи кластеризацію (розбиття на класи поблизу)за деякими важливими ознаками), класифікацію (віднесення до певних заданих класів), верифікацію (виявлення конкретного шуканого об'єкта);- Визначення характеристик тих з виявлених об'єктів,які необхідні виконання роботом конкретних завдань .При вирішенні цих завдань СКЗ також часто комплексуються з іншими сенсорними системами. Крім того, як зазначено вище, СКЗ застосовуються для візуалізації вихідної інформації інших типів сенсорних систем.

СКЗ можуть бути:

– одномірними (лінійка), двох та тривимірними;

– монохромними (напівтоновими, чорно-білими);

– кольоровими.

Обробка здійснюється спеціальним пристроєм введення (цифрова плата з пам'яті-зображення). Для підвищення швидкодії особливо при роботі з досить складними зображеннями переходять до багато-процесорним системам з поділом завдань на підзадачі, які можна вирішувати паралельно. Існують СКЗ і з послідовною(Конвеєрної) структурою. Вони застосовуються для обробки більших масивів даних за тривалий період часу. У комп'ютерах використовується цифрове зображення, яке являє собою отриману від датчиків зображення совокупність рядків і стовпців пікселів (pixels - скорочення слів picture element - елемент зображення), попередньо змінених у цифрову форму. Кожен піксель характеризується інтенсивністю (яскравістю), що є числом. Зазвичай використовують однобайтові (8-бітові) числа, що дають числові значення від 0 до 255, рідше - 10 бітові (1024 значення).Датчик зображення – це оптоелектронний перетворювач (ОЕП), що працює в діапазоні видимого світла на рис. 1.5.

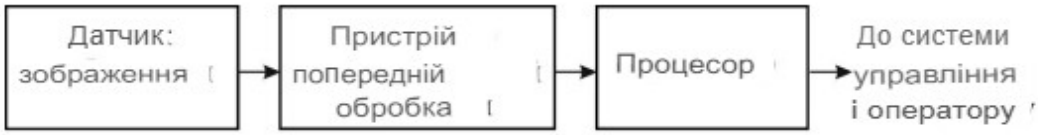

Рисунок 1.5 – Схема системи технічного бачення

Так само згадані вище ОЕП, що працюють в інших діапазонах електромагнітного випромінювання до видимого світла - ІЧ та радіо після - УФ, рентгенівське та гама випромінювання.

Перетворення видимого світла є телекамерою одного здвох типів - у вигляді вакуумних електронно-оптичних приборів, які історично з'явилися першими, і більш тимчасових твердотільних камер на основі осередків (ПЗС) або фотоелементів (фотодіоди, фототранзистори, фоторезистори).

Основні характеристики цих датчиків:

– роздільна здатність (дозвіл по вертикалі та по горизонталі);

– чутливість (мінімальна сприймана освітленість);

– спектральна характеристика (діапазон частот регістр електромагнітних коливань).

Зображення можуть виходити так само і за допомогою сканування. далекомірів - локаторів.

Відповідно до зазначеного вище функціонального складу у СКЗ послідовно вирішуються такі завдання обробки зорової інформації:

– попередня обробка відеозображення, що отримується від датчика, у вигляді його фільтрації з метою підвищення якості зображення шляхом звільнення від шумів та інших перешкод, включаючи зміну освітленості та тіні, згладжування, підвищення контрастності з посиленням меж об'єктів та їх частин, преображення аналогового сигналу в цифровий;

– сегментація, тобто декомпозиція зображення з послідовним виділенням окремих об'єктів із загальної картини, потім їх частин і т. д. (виявленням контурів, ділянок певної текстури та кольори);

– визначення характеристик цих об'єктів, тобто. ознак (дескрипторів) для їхнього подальшого розпізнавання (виділення відрізків прямих, отворів, округлостей, визначення величини периметра, площі тощо);

– розпізнавання, тобто кластеризація, класифікація або верифікація. При цьому розпізнавання можливе і за неповної видимості частини об'єктів.

У цьому випадку висувається і перевіряється кілька гіпотез. Останнє може вимагати і додаткового огляд об'єкта.

При тривимірних зображеннях об'єктів вони зазвичай розпізнаються за двовимірними проекціями, а третя координата використовується визначення взаємного розташування об'єктів.

Разом з тим існують методи та об'ємного розпізнавання.

Зокрема,відомий метод суміщення 3D зображень із 3D геометричними моделями, утвореними з примітивів типу циліндр, паралелепіпед і т. п. (Тут суміщення проводиться вже за трьома характерним точкам.)Обчислювальні методи обробки зображень розвиваються вже протягом понад 30 років.

Спочатку використовувалися великі ЕОМ, потім із появою персональних комп'ютерів вони стали основним апаратним забезпеченням цих методів.

У наш час існують десятки програм обробки зображень та розпізнавання образів, у тому числі осіб, відбитків пальців,рукописних текстів, номерів автомашин, штрих-кодів. Багато їх працюють у реальному масштабі часу.

Як правило, вони передбачають попередній етап навчання (з учителем або самонавчання).

До них насамперед відносяться стиснення інформації та її зберігання .

Щодо останнього, то для вирішення задачі розпізнавання необхідна база даних для зображень еталонів та прототипів,класифікованих за характерними геометричними та словесними ознаками.

Зауважимо, що більшість способів та прийомів обробки зображень у СКЗ аналогічні існуючим у живій природі та підказані фізіологами та психологами.

Огляд видимої зони, виділення окремих об'єктів, їх кластеризація, класифікація або верифікація, розподіл геометричних та оптичних параметрів зовнішнього середовища та її об'єктів, комплексування з іншими сенсорними системами. для визначення різних фізичних і хімічних параметрів. зовнішнього середовища та її окремих об'єктів, візуалізація інформації з інших сенсорних систем та інших джерел.

Ці функції визначають призначення СКЗ в системах робототехніки у вигляді забезпечення візуальною інформацією систем автоматичного управління та операторів систем автоматизованого управління засобами робототехніки для виконання різних технологічних та інших операцій, що вимагають візуальної інформації (моніторинг зовнішнього середовища, охорона, бойові операції, контроль якості виробів, що виготовляються, сервісне обслуговування людей тощо).

1.5 Висновки до першого розділу

В розділі 1 проводиться огляд комп'ютерних робототехнічних систем різного типу, аналіз існуючих фізичних та графічних інструментів моделювання елементів комп'ютерних робототехнічних систем та маніпуляторів.

Проаналізовано робототехнічних системи та наведена їх класифікація.

Розглянута основна інформація про існуючі комп'ютерно робототехнічні системи, які використовуються у фармацевтиці.

Проведено аналіз Систем технічного зору та наведено приклад їх використання в мобільних роботах.

2 АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ FESTO ROBOTINO

2.1 Аналіз пристрою та програмування мобільного робота

Robotino - це мобільний робот, що пересувається на трьох роликових коліс типу "omni wheel" ("omnidirectional" - всеспрямований).

У рух робота приводять три двигуни постійного струму, осі яких розташовані під кутом 120 градусів один до одного.

Завдяки такій конструкції робот може переміщуватися у всіх напрямках, а також повертатися навколо своєї осі.

Оскільки Robotino призначається, головним чином, навчання, він виконаний модульно; всі технічні компоненти (електроприводи, датчики, камера) можна відключити від робота і вивчити окремо (рис. 2.1).

Рисунок 2.1 – Загальний вигляд мобільного робота

Корпус робота – сталевий корпус з бампером, що забезпечує легкий доступ до батарей, двигунів та портів робота.

Підсистема живлення – дві акумуляторні батареї (12 В / 5 А·год) кислотно-свинцем (lead-acid) або 12 В / 9 А·год), що дозволяють роботу працювати в автономному режимі кілька годин.

Двигуна підсистема – три двигуни постійного струму, редуктор та ремінна передача на роликові колеса, що дозволяють роботу рухатися у різних напрямках.

Для стабілізації швидкостей обертання валів двигунів робота використовуються вбудовані ПІД-регулятори з встановленими коефіцієнтами.

Підсистема одометрії – інкрементні енкодери на валах двигунів.

Підсистема введення/виводу – плата, що виконує комунікаційний зв'язок між комп'ютером робота та його датчиками, двигунами та інтерфейсом введення/виводу.

Підсистема бездротового зв'язку із зовнішнім керуючим комп'ютером (Wi-Fi точка доступу).

Бортовий обчислювач – одноплатний промисловий комп'ютер формату PC/104+ (має шини ISA та PCI) із процесором на 300 МГц.

Як пам'ять використовується картка Compact Flash (1024 Мб), на яку встановлена вбудована версія операційної системи Linux.

Різні датчики робота - 9 інфрачервоних датчиків відстані (на бампері), USB-камера рис. 2.2 .

Керувати роботом можна як за допомогою програм, записаних у пам'ять робота, так і дистанційно - по каналу бездротового зв'язку Wireless LAN (WLAN). Крім того, можливе автономне програмування робота, через підключені до нього монітор та клавіатуру.

У комплекті з роботом йдуть програми Robotino View та Robotino Sim, призначені для інтерактивного графічного програмування робота рис. 2.3.

Рисунок 2.2 – Підключення додаткового обладнання до Robotino

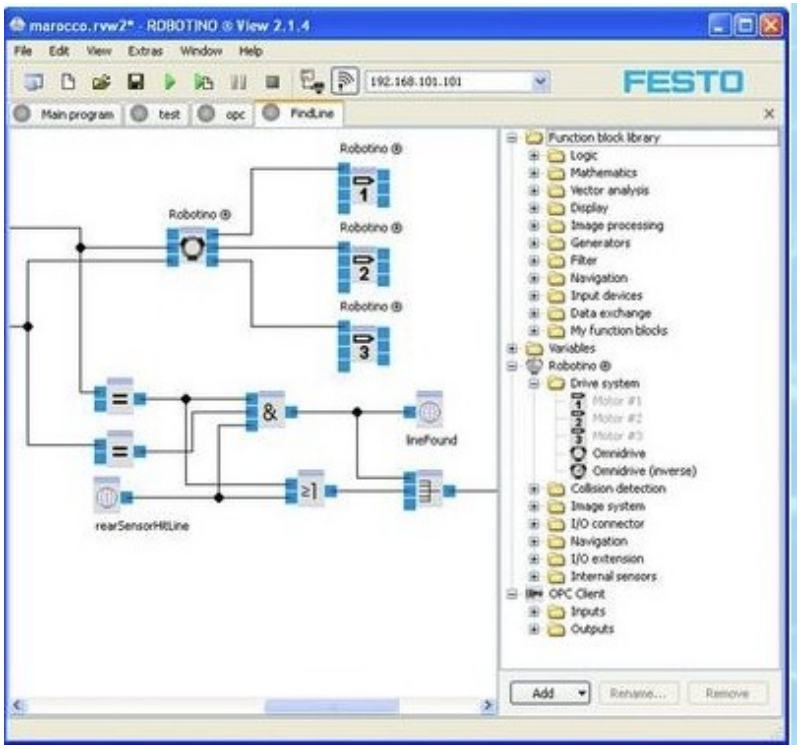

Рисунок 2.3 – Блок підпрограммы Robotino View

2.2 Можливості сенсорної системи Robotino

Аналіз складу сенсорної системи сенсорного сенсорного сенсорного датчика включає датчики вимірювання інфрачервоних відстань, модуль гіроскопа, різні оптоелектронні датчики та веб-камера.

Для визначення перешкод Роботіно використовує інфрачервоні датчики відстані, які показують відстань до об'єкта перешкод. Такі датчики в роботіно існують дев'ять (IR1 - IR9), розташоване навколо периметру нижньої бази робот-корпусу (рис. 2.5). Тип датчика - інфрачервоний дистанційний вимірювач гострий GP2Y0A21YK з вимірювальним кордоном від 10 см до 80 см.

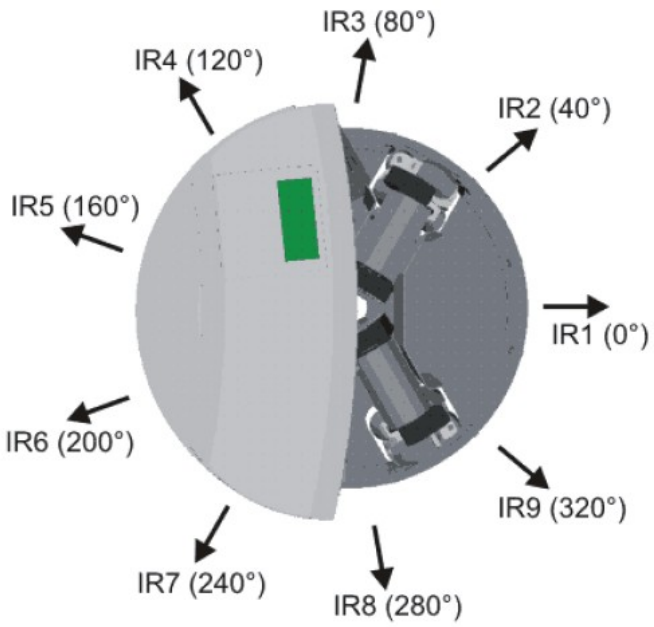

Рисунок 2.4 – Розташування датчиків відстані

Для вимірювання відстані до об'єкта існують оптичні датчики, що працюють у методі триангуляції. Найбільш поширеними з них працюють на довжині хвилі, інфрачервоних (ІЧ) датчиків відстані з вихідною аналоговою напругою, виробленою гострим. Різкі датчики мають IR світлодіодну з лінзою, яка випромінює вузький світловий промінь, який відбивається від

перешкоди, поглинається фотоприймачем. Зареєстрована провідність перетворюється на напругу, а після аналого-цифрового перетворення відповідно до зареєстрованих даних, ви можете обчислити відстань. Точна діаграма між відстані та виходом наведена в специфікації датчика (рис. 2.5). У датчиках, за його типом, існує границя вимірювання, в межах яких вимірювання датчика забезпечують оголошену надійність. Мінімальна виміряна відстань обмежена особливостями різкого датчика, а саме, вихідна напруга на певній відстані починає різко падати з зменшенням відстані. Це означає, що одне значення вихідної напруги відповідає двом можливим відстаням, які можуть служити як провал алгоритму керування. Однак з експериментальним тестуванням сенсорної системи робота зіткнулася з ситуацією, коли кожен датчик від цілісних системних питань значень, які значною мірою порушуються заявленими значеннями. Це особливо виражена при зміні освітлення зовнішнього простору робота.

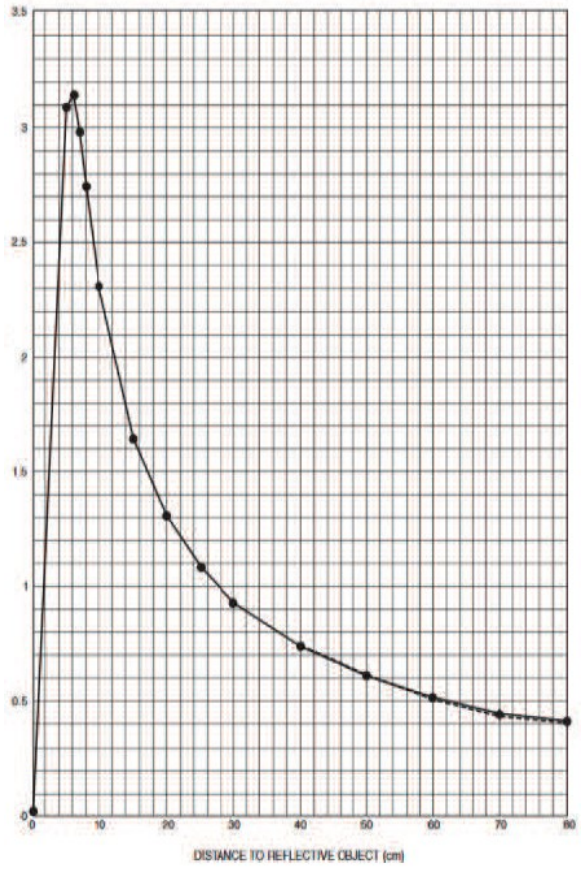

Рисунок 2.5 – Залежність напруги від відстані

2.3 Вплив рівня освітлення на показання далекоміра

Ряд вимірювань проводився за різними середовищами на основі робот, тоді як вимірювання ІЧ-датчиків були записані. Вимірювання проводилися для двох рівнів освітлення - 0 LCS (у темряві) та 250 лк (ефір лабораторне освітлення). Вимірювання показань були зроблені на відстані до перешкоди, кілька 50 мм у діапазоні від 0 мм до 400 мм. Результати вимірювань наведені у вигляді графіків залежності (рис. 2.6, рис. 2.7). Метод найменших квадратів залежать від залежності та математичних моделей залежності від відстані від вимірюваного значення напруги для нормального рівня освітлення зовнішнього простору робота

$$y = 12.264 \cdot x^{-1.112}, \tag{2.1}$$

в умовах недостатнього освітлення

$$y = 13.67 \cdot x^{-1.131}, \tag{2.2}$$

де y – оцінка відстані до об'єкта перед датчиком; x – показання датчика, що реєструється.

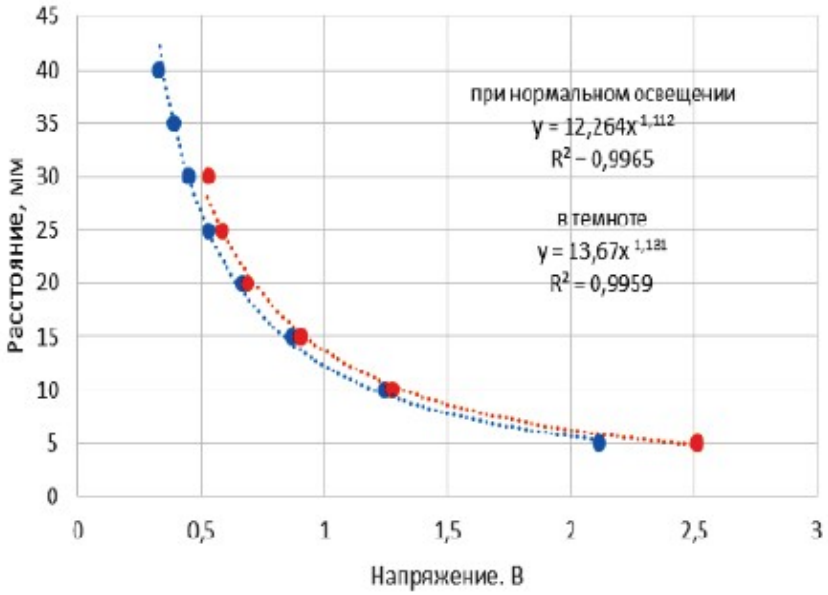

Рисунок 2.6 – Залежність відстані від показників далекоміра [12]

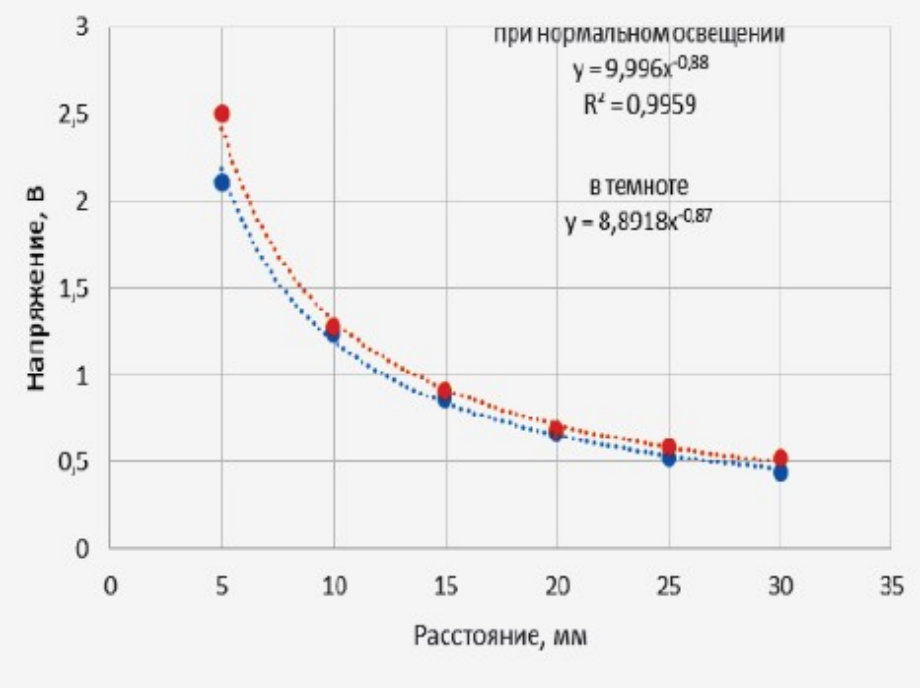

Рисунок 2.7 – Залежність показань далекоміра від відстані [12]

Моделі забезпечують високий рівень детермінації (R = 0,99). Порівнюючи неузгодженість показань ІЧ датчика у двох режимах освітленості можна побачити, що значні відхилення в діапазоні від 50 мм до 100 мм, далі помилка різко змінюється (відносна помилка на рівні 1,2 %) та монотонно збільшується (до 16 %) зі збільшенням відстані до перешкоди. Залежність помилки вимірювання за різних рівнів освітленості наведено на рис. 2.8.

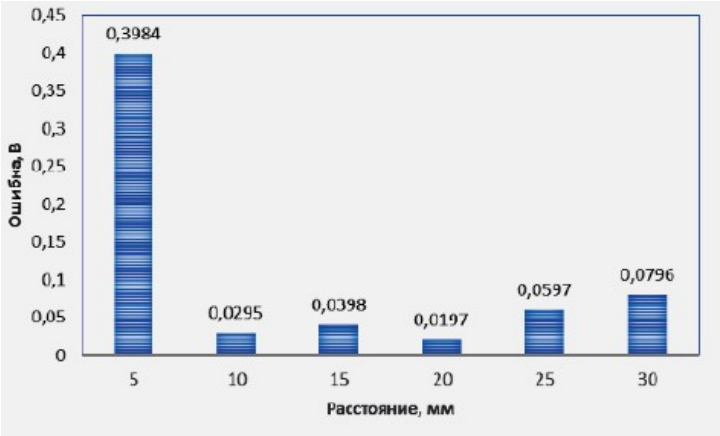

Рисунок 2.8 – Залежність помилки вимірювання за різних рівнів освітленості [12]

Приклади задач FESTO Robotino, що вирішуються автономно з використанням сенсорної системи

На базі платформи FESTO Robotino можуть бути реалізовані різні за складністю алгоритми розв'язання різноманітних завдань: стеження та пошук об'єкта за кольором, формою, пошук виходу з лабіринту тощо.

Так, стеження та пошук об'єкта за кольором може бути реалізований з використанням монтується на платформу камери та алгоритму, побудованого на визначенні за зображенням позиції центру та обчислення площі об'єкта.

При зміщенні об'єкта щодо координати x алгоритм управління виробляє управляючі впливи для двигунів і за рахунок переміщення вліво/вправо центрує об'єкт щодо області камери, за рахунок переміщення вперед/назад досягає заданого значення площі об'єкт на зображенні (рис. 2.9). Алгоритм управління для вирішення задачі стеження за об'єктом за кольором включає два контури управління:

– основний контур управління побудований на аналізі зображення з камери і забезпечує основну функцію (рис. 2.10);

– локальний контур управління побудований на основі показань ІЧ датчиків відстані та забезпечує своєчасну аварійну зупинку.

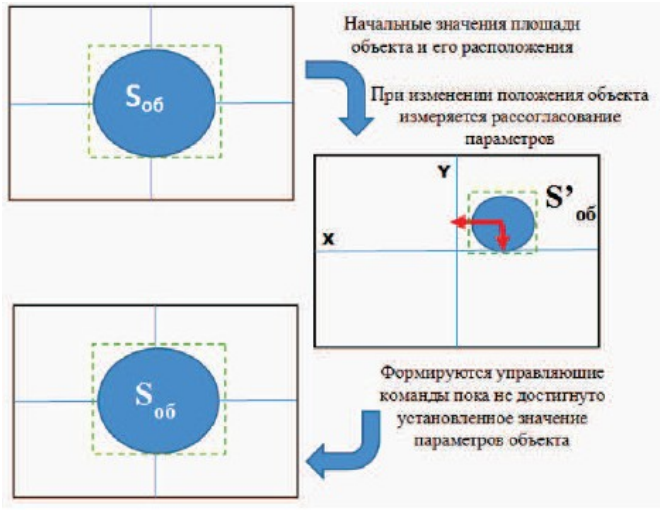

Рисунок 2.9 – Схематичне пояснення принципу керування [13]

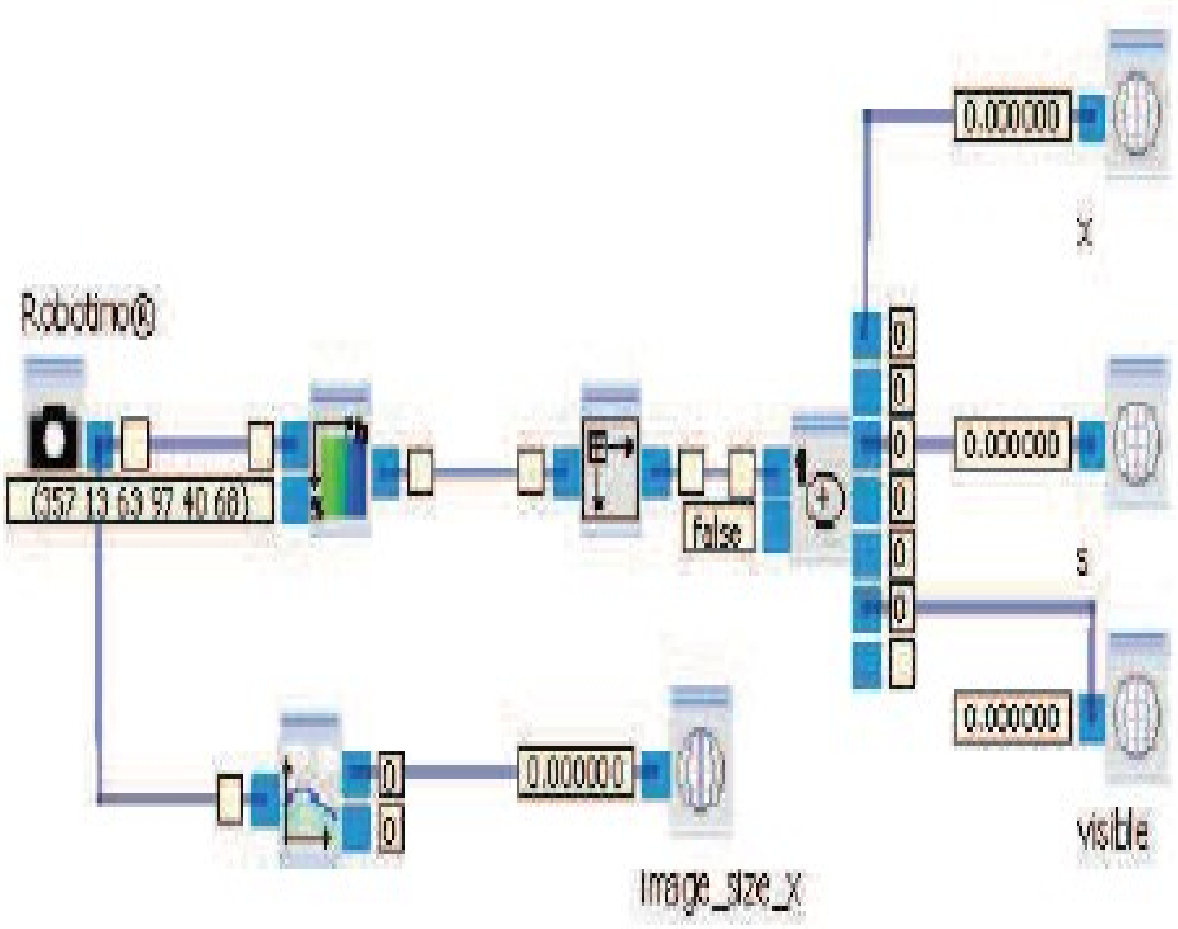

Рисунок 2.10 – Програма управління RobotinoView [19]

2.4 Висновки до другого розділу

Проведений аналіз сенсорної системи роботизованої мобільної платформи FESTO Robotino дозволив виявити її обмеження, які знижують якість управління за умов автономного вирішення запрограмованих завдань.

У ході досліджень встановлено залежності, які можуть послужити основою для побудови алгоритмів управління роботом як при нормальному освітленні, так і в умовах недостатньої освітленості.

Перспективи подальших досліджень пов'язані з використанням виявлених можливостей та обмежень для побудови алгоритмів автономної роботи робота.

3 РОЗРОБКА АЛГОРИТМІЧНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ МОБІЛЬНОГО РОБОТА FESTO ROBOTINO

3.1 Програмування функцій керування мобільним роботом

Було виконано ряд алгоритмів програмування мобільного робота Robotino на платформі RobotinoView . У ході ознайомлення я зрозумів, що такі робототехнічні системи мають широкий спектр можливостей, а блокове програмування робить процес простим, зрозумілим та досить цікавим. Наведу кілька прикладів таких алгоритмів.

3.1.1 Програмування джойстика

Одним із комплектуючих Robotino є джойстик. Для його використання нам необхідно призначити певним кнопкам джойстика певні дії робота. Найпростіший варіант рух по одній осі (рис 3.1).

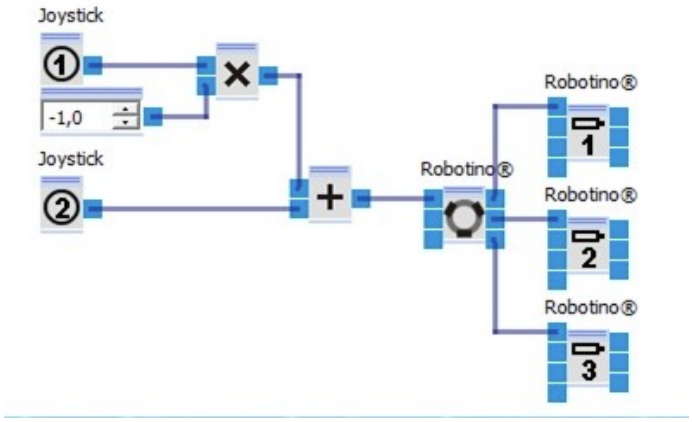

Рисунок 3.1 – Програмування кнопок джойстика для переміщення по осі X

Для забезпечення руху робота в прямому (кнопка 2) та зворотному напрямку (кнопка 1) по координатній осі X, можливе рішення (задавати рух у

зворотному напрямку), представлене на рис. 3.1, де сигнал від кнопки 1 множиться на значення заданої константи (-1) .

3.1.2 Програмування робота за допомогою блоків одометрії

Якщо нам необхідний рух робота за певними координатами, ми можемо скористатися блоками одометрії. Суть полягає у завданні конкретних координат, на які робот зміститься щодо початкового положення (рис. 3.2).

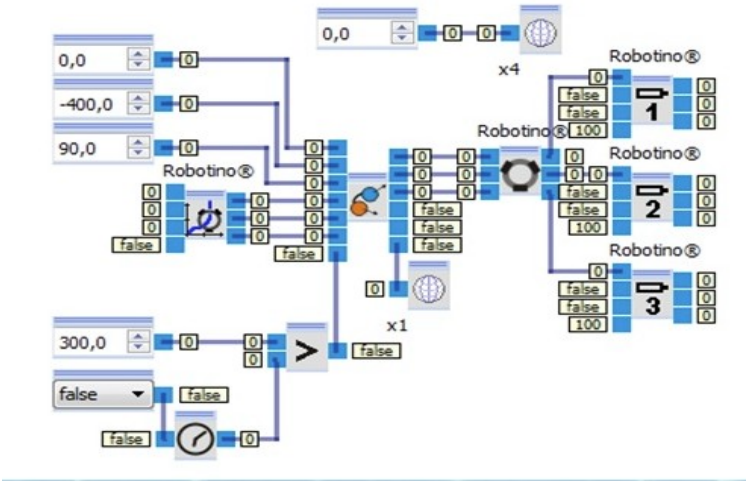

Рисунок 3.2 – Приклад програмування робота з використанням блоків одометрії

Цей алгоритм визначає лише одне переміщення. Для послідовного переміщення у різні координати необхідно створити кілька підпрограм. Всі підпрограми подібні, різниця лише в заданні координат і контролі зв'язуючих змінних, тобто беручи цю програму за зразок, можна задати будь-яку просту траєкторію руху робота. У кожній наступній підпрограмі обнулюється зв'язуюча змінна.

У процесі роботи з даною робототехнічною системою було розроблено алгоритм для вирішення двох завдань – розпізнавання кольорів у просторі та рух до певних сегментів; рух робота по лінії.

У процесі пошуку відповідей ми пройшли етап самонавчання методом спроб і помилок. У результаті шукане рішення знайшли. Робот по черзі знаходив необхідні кольори і рухався в їхньому напрямку, попутно центруючись, аж до спрацьовування переднього датчика, який посилав сигнал на зупинку. Також було сформовано цикл, що дозволяє повторювати програму постійно.

3.1.3 Знаходження заданих кольорів у просторі Robotino View

Мета: вивчити нові функціональні блоки та елементи середовища Robotino View. Скласти програму для автоматичного переміщення мобільного робота за заданими кольорами.

Обладнання: мобільний робот, робочий комп'ютер із встановленими програмами Robotino View та Robotino SIM.

У цій роботі необхідно створити програму, яка дозволить роботу по черзі шукати та переміщатися за заданими кольорами. Для цього нам потрібні такі графічні блоки (рис. 3.3 – рис. 3.5):

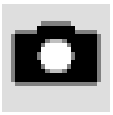

Рисунок 3.3 – (Robotino/Image system/Camera) – блок, що підключає камеру робота

Рисунок 3.4 – (Function block library/Image processing/Segmenter) –блок, порядковий номер, що присвоює вибраним кольорам

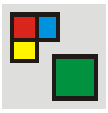

Рисунок 3.5 – (Function block library/Image processing/Segment Extractor) – блок пошуку поточного кольору сегмента та отримання даних про координати його центру

На рис. 3.6 – (Function block library/Image processing/Image Information) блок видачі ширини та висоти прийнятого зображення з камери робота.

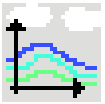

Рисунок 3.6 – (Function block library/Image processing/Image Information) – блок видачі ширини та висоти прийнятого зображення з камери робота.

На етапі search (рис. 3.7) робот проводить пошук поточного кольору сегмента, здійснюючи обертальний рух із заданою кутовою швидкістю.

На рис. 3.8 – структура кроку пошуку кольору сегмента (search).

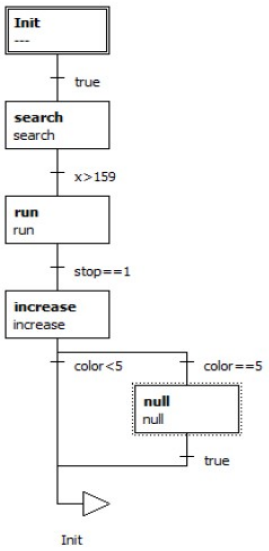

Рисунок 3.7 – Блок-схема головної програми

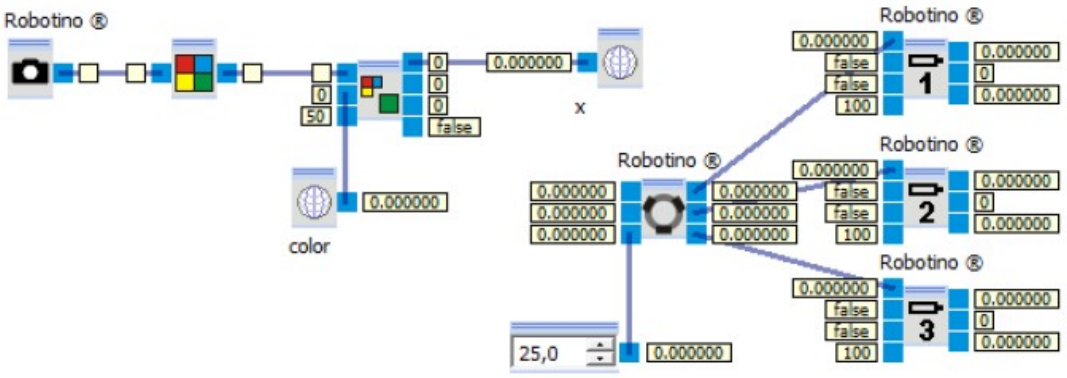

Рисунок 3.8 – Структура кроку пошуку кольору сегмента (search)

У блоці Segmenter (рис. 3.9) при включеній симуляції надаються порядкові номери кольорів усіх сегментів. Для цього під час обертання

робота при попаданні в область видимості камери потрібного кольору ставиться прапорець Freeze image, потім колір виділяється мишкою і додається до директиви кнопкою '+'. Аналогічно для інших кольорів.

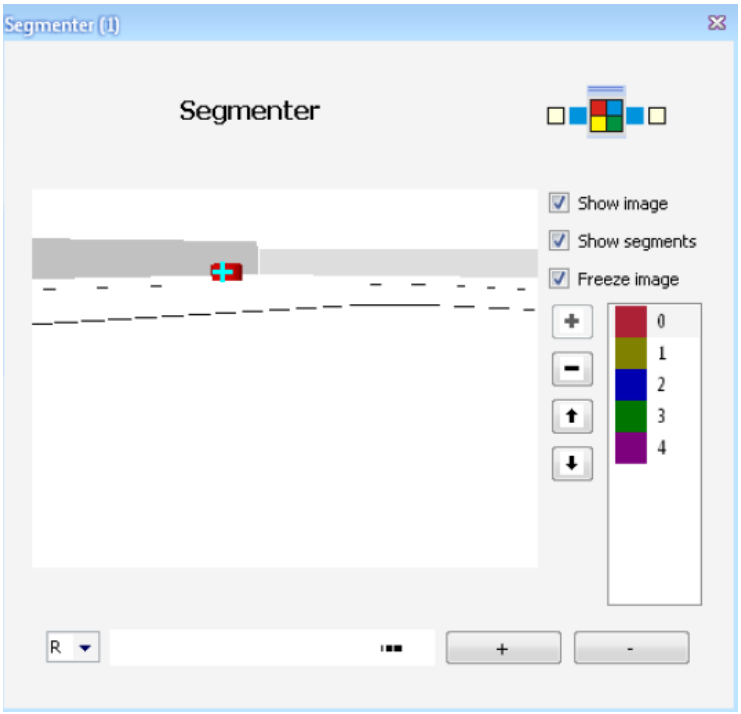

Рисунок 3.9 – Вміст блоку Segmenter

Зображення знайденого сегмента з поточним порядковим номером кольору підводиться до Segment Extractor (рис. 3.10) сигналу Input (Image).

Рисунок 3.10 – Опис вхідних та вихідних сигналів блоку Segment Extractor

Input (image) – зображення, що входить із Segmenter; Selected segment (int) –порядковий номер поточного кольору сегмента; Minimum area (int) – мінімальна площа сегмента виявлення; x (int) - поточна координата положення центру сегмента по осі X; y (int) - поточна координата положення центра сегмента по осі Y; Area (int) – поточна площа сегмента; Segment found (bool) – логічна змінна, яка повідомляє, що сегмент знайдений.

Канал Selected segment зчитує поточний порядковий номер сегмента зі змінною color.

Поточна координата положення центру сегмента по осі X записується в змінну x.

На наступний крок run програма переходить за умови, якщо x > 159, де 159 - це половина ширини зображення з камери (у нашому випадку 320 пікселів). Для камер з іншим розширенням ця величина визначатиметься за формулою

$$\frac{ширина\ зображення\ (в\ пікселях)}{2} - 1. \qquad (3.1)$$

У кроці run (рис. 3.11) константою визначається швидкість руху по осі X і зіставляється центр зображення камери з центром поточного сегмента, регулюючи тим самим положення робота щодо осі Y.

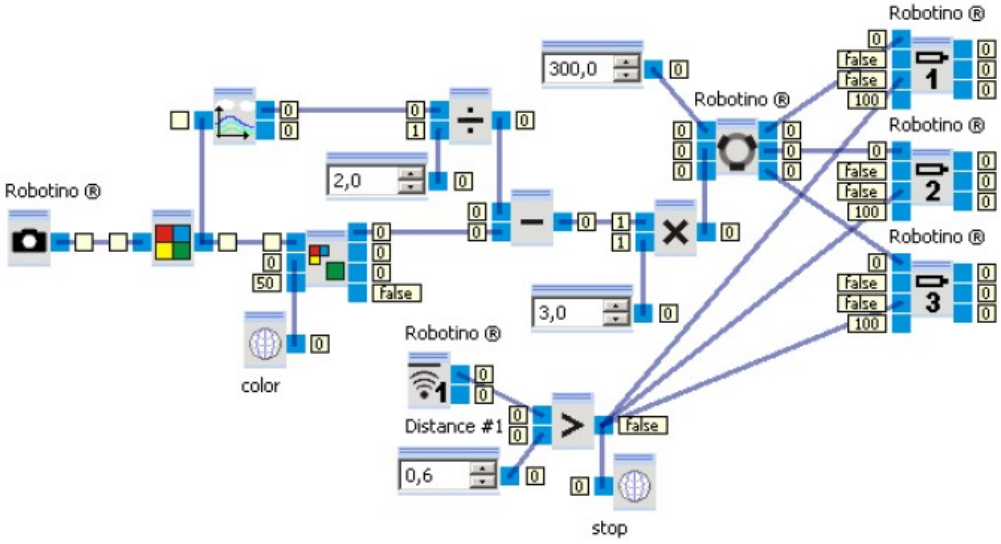

Рисунок 3.11 – Структура кроку руху робота до поточного кольору сегмента

У міру наближення робота до об'єкта на датчик надходить сигнал. Коли сигнал з датчика перевищує задану константу, результат порівняння записується змінну stop. Умовою переходу на наступний крок є рівність це змінної значення 1 (true). У кроці increase (рис. 3.12) до поточного значення color додається одиниця, тим самим змінюючи колір, що шукається.

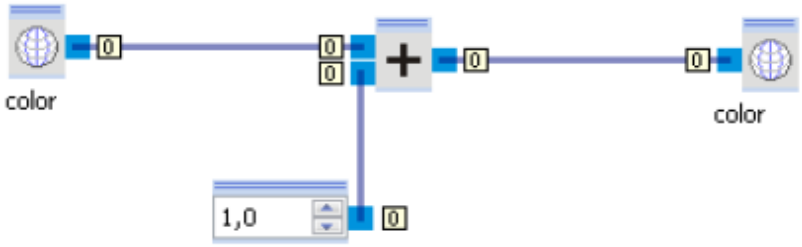

Рисунок 3.12 – Структура кроку зміни кольору сегмента, що шукається

Потім у main program відбувається розгалуження за двома умовами. Якщо порядковий номер кольору менше 5 (число заданих кольорів), програма повертається на початок і виконує вищезгадані кроки для наступного кольору.

Якщо порядковий номер кольору дорівнює 5, програма потрапляє на крок null (рис. 3.13), який обнуляє змінну color. Потім програма здійснює повторний пошук початкового кольору.

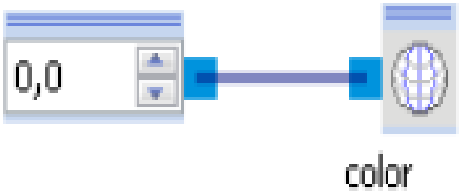

Рисунок 3.13 – Структура кроку обнулення кольору сегмента

Діалогове вікно візуалізує сигнали на каналах. Для кожного каналу можна змінити налаштування, наприклад посилення, напр, також можна деактивувати окремі канали.

На рис. 3.16 – детектор ліній шукає лінії певного кольору у вхідному зображенні

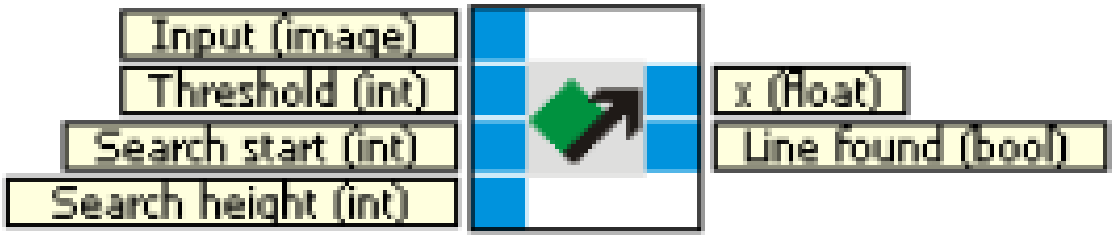

Рисунок 3.16 – Блок детектор ліній

На рис. 3.17 область пошуку лінії позначена горизонтальними червоними лініями. Нижній рядок позначає «Початок пошуку». У верхньому рядку позначки "Початок пошуку" + "Висота пошуку".

Червоний + позначає темний або світлий край лінії, якщо дивитися зліва направо.

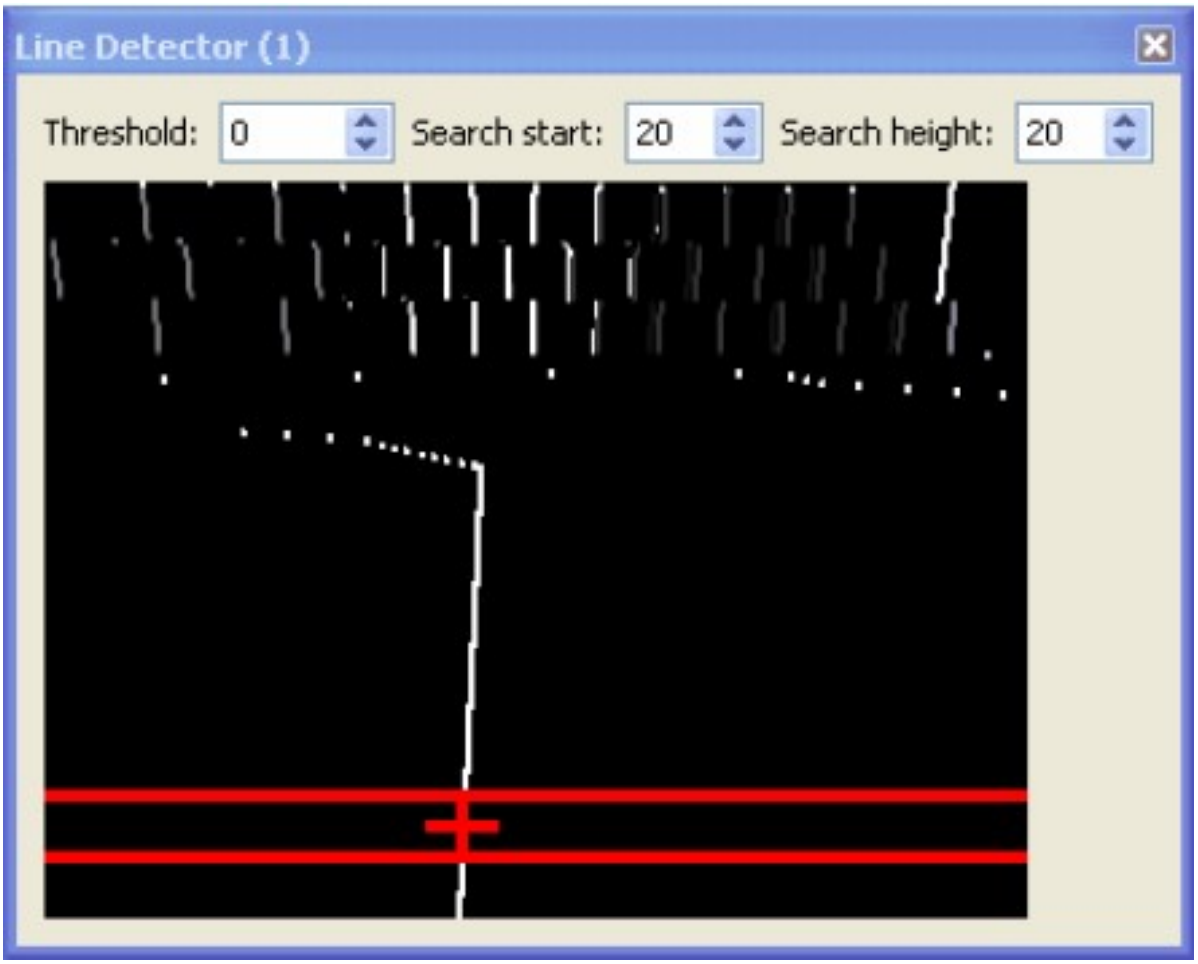

Рисунок 3.17 – Діалогове вікно детектор ліній

Зображення з камери Robotino (із симулятора Robotino) використовується як вхід для рядка детектор.

Ми використовуємо функціональний блок Image Information, щоб відобразити положення x лінії від діапазона [0, ширина зображення] до [-image width/2, image width/2], що в нашому випадку [-160,160]. Масштаб

функціональний блок використовується для перемикання знака та для масштабування результату функції віднімання блокувати.

Значення можна використовувати безпосередньо для повороту Robotino, щоб Robotino повернув праворуч, якщо лінія знаходиться праворуч і повертає ліворуч, якщо лінія розташована зліва.

З постійною швидкістю вперед Robotino слідує за лінією на рис. 3.18 – діалогове вікно детектор ліній.

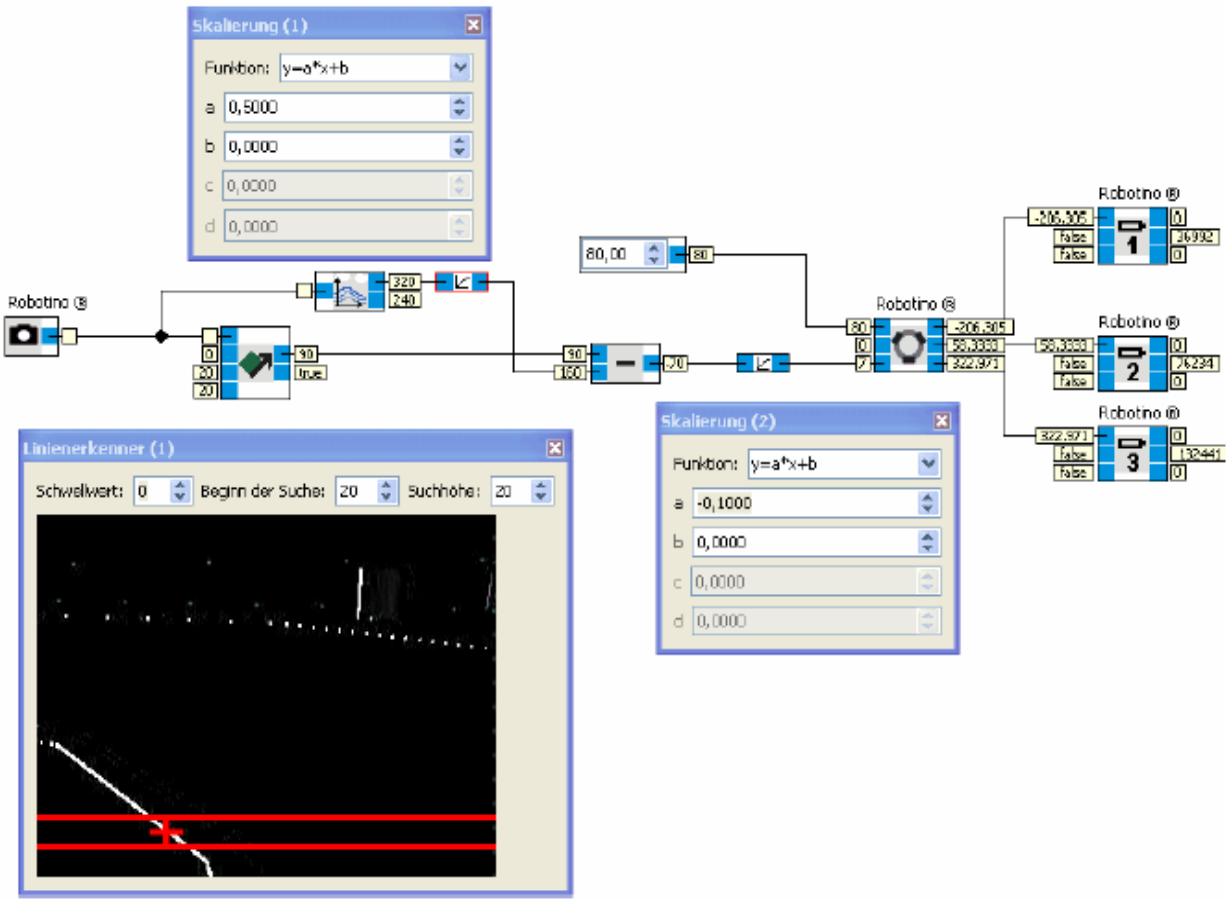

Рисунок 3.18 – Діалогове вікно детектор ліній

Програма зчитування зображень генерує тестову послідовність зображень.

Детектор нижньої лінії використовує ціле зображення, тоді як верхній детектор рядків шукає лише ROI(Регіон інтересів) на рис. 3.19 – діалогове вікно детектор ліній.

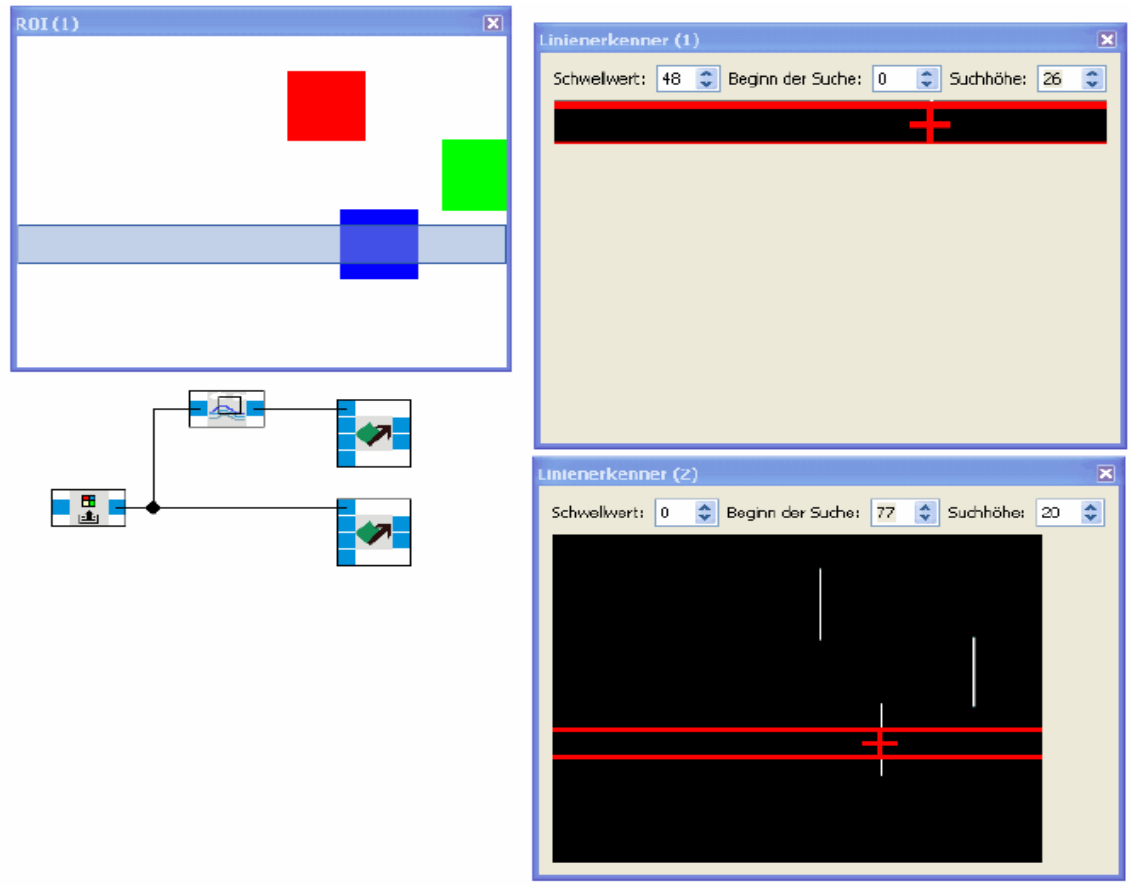

Рисунок 3.19 – Діалогове вікно детектор ліній пошук регіон інтересів

У діалоговому вікні перетворення колірного простору можна вибрати цільовий колірний простір (рис.3.20 – рис.3.21).

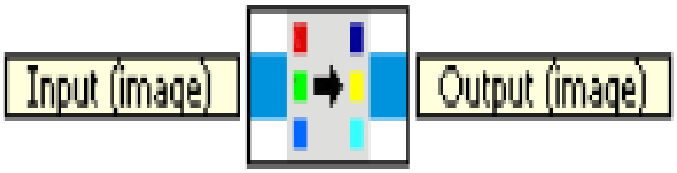

Рисунок 3.20 – Програмний блок перетворення колірного простору

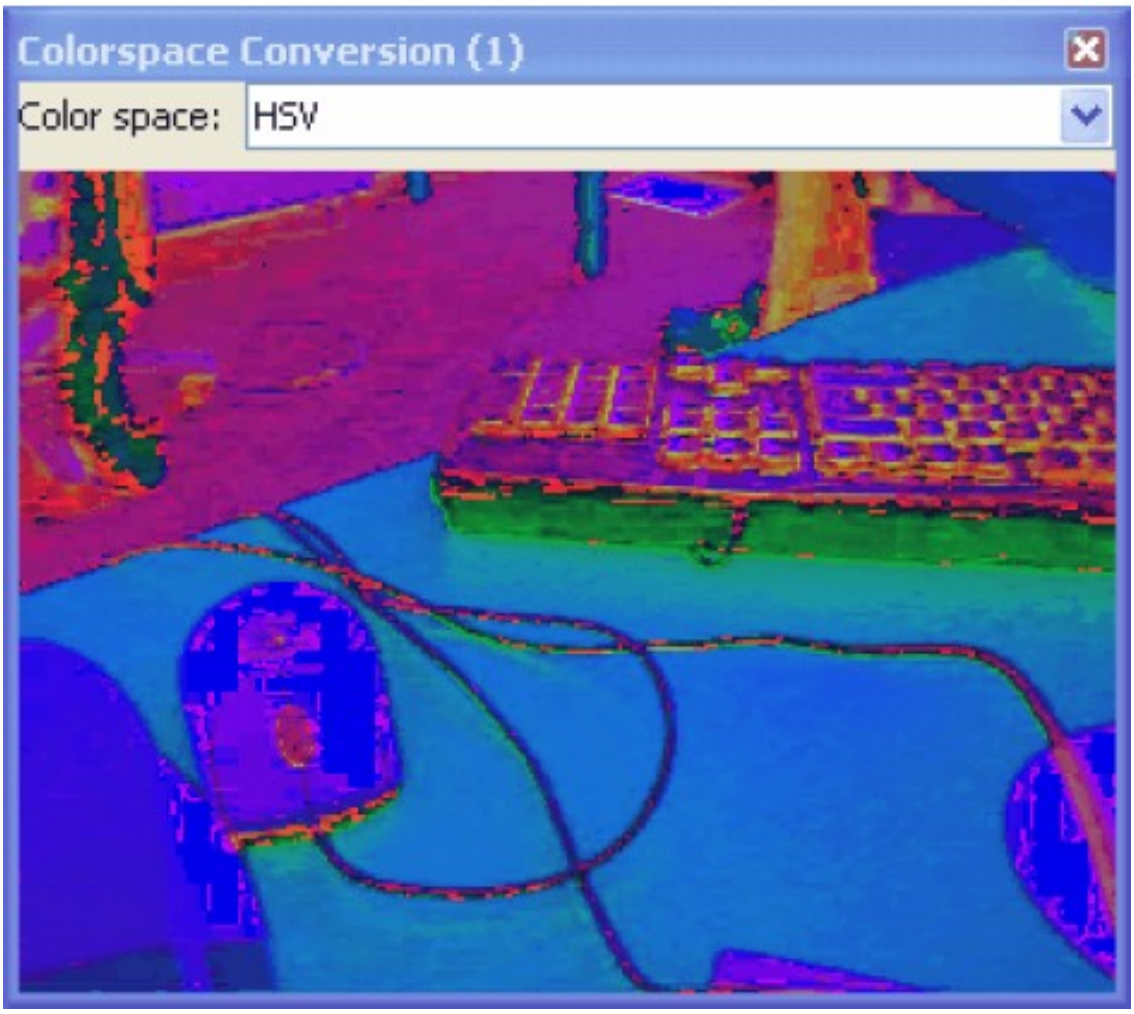

Рисунок 3.21 – Діалогове вікно перетворення колірного простору

3.1.4 Визначення відстані до перешкоди

Robotino View це інтерактивне візуальне програмно-навчальне середовище для робота, яке з'єднується за допомогою Wireless LAN прямо з системою управління робота.

Тому, щоб керувати Robotino, потрібен лише персональний комп'ютер, який може встановити зв'язок із WLAN (рис.3.14).

Рисунок 3.14 – Порт для з'єднання Robotino з комп'ютером

Після встановлення зв'язку Robotino буде точкою доступу для комп'ютера. Кожен Robotino повинен мати однакову IP-адресу, тому що кожен формує свою власну мережу.

У ході виконання роботи важливими складовими є 9 інфрачервоних датчиків, розташованих з боків бампера та датчик захисту від зіткнень.

Інфрачервоний датчик відстані складається з емітера, який випромінює промінь інфрачервоного світла, відповідного приймача та електронного обчислювального (оцінного) блоку (рис.3.15).

Рисунок 3.15 – Інфрачервоний датчик відстані

Інфрачервоний датчик відстані містить в собі передавач та приймач в інфрачервоному діапазоні.

На відміну від ультразвукового датчика відстані не вказує відстань до перешкоди, а просто показує чи є перешкода в зоні видимості.

При спрацьовуванні датчика вихідна напруга стає рівним 0.

Відстань до об'єкта рецептори визначають за допомогою тріангуляції (рис.3.16).

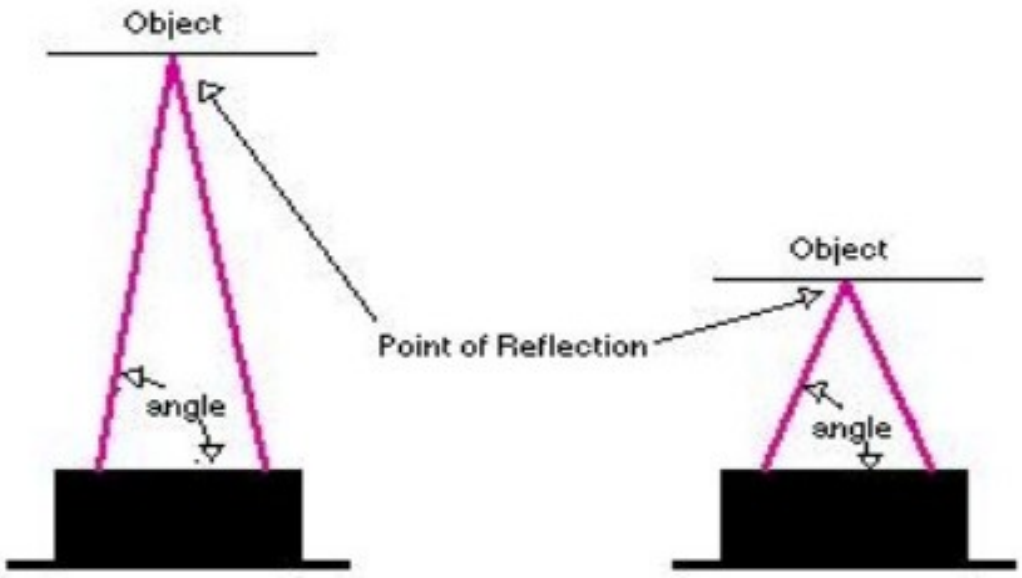

Рисунок 3.16 – Тріангуляційний метод вимірювання

Випромінювач випромінює інфрачервоний промінь. Якщо цей промінь не потрапляє на предмет, він не відбивається і тому приймач не сприймає промінь світла.

Проте, якщо світло відбивається від предмета, промінь світла виявляється у межах певної області приймача.

Так як передавач випромінювання і фотоприймач розташовані на малій відстані один від одного в межах датчика, промені світла, що випускаються і приймаються, формують трикутник.

Залежно від відстані промінь світла потрапляє різні ділянки приймача. Приймач складається із світлочутливого детектора (PSD), який визначає різні

точки падіння променя. Блок обробки сигналів перетворює в аналогову величину напруги.

PSD є фотодіодом пластинчастої форми.

Він складається з металевого та світлочутливого шарів. На краях цих шарів розташовані металеві електроди, якщо промінь світла потрапляє в крапку на цьому світлочутливому шарі, це роз'єднує носії заряду, які породжують електричний струм у напрямку двох електродів.

Неосвітлена частина шару діє як опір.

Залежність між струмами залежить від кількості падаючого світла; тому вимір відстані залежить від коефіцієнта відображення та матеріалу об'єкта (рис.3.17) [3].

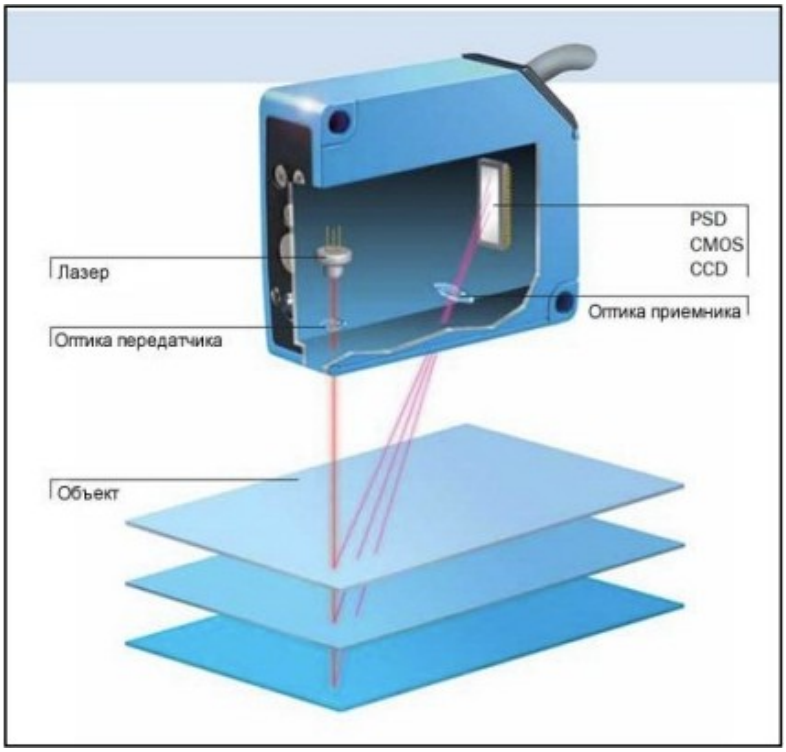

Рисунок 3.17 – Визначення відстані до об'єкту

Для позбавлення від можливих перешкод сенсори випромінюють інфрачервоний сигнал з модульованою частотою. Це дозволяє практично повністю застрахуватися від завад від навколишнього світла. Крім того,

датчики показують майже повну байдужість до кольору об'єкта виявлення (датчик здатний виявляти чорні стіни за сонячного світла [2].

У більшості випадків випромінювач, приймач та блок обробки сигналів об'єднані в один пристрій.

Приклади типових областей застосування інфрачервоних датчиків: системи керування відстанню під час паркування на автомобілях, відчинення дверей або в системах аварійної сигналізації.

Датчик захисту від зіткнень Robotino складається з так званої чутливої кромки.

Ця чутлива кромка складається з полімерного профілю змінної форми з інтегрованою комутаційною порожниною.

Дві роздільні провідні області розташовані в межах порожнини, які коротко змикаються при впливі тиску на чутливу кромку, таким чином, генеруючи сигнал для блоку оцінки.

Чутлива кромка на Robotino працює у відповідність до принципу струму в робочій точці так, що розрив кабелю може бути виявлений і Robotino зупиниться.

Буфер, доступний у програмі Robotino View в Robotino, знаходиться в папці апаратних засобів і не потребує параметризації.

Він виробляє один сигнал після дотику та використовується головним чином для зупинки Robotino у разі зіткнення.

Для цього він під'єднаний до виходу папки Sequence Control (Контроль послідовності).

Тоді у разі зіткнення програмна послідовність переривається.

Чутливі грані в основному використовуються в безпечній технології, щоб убезпечити людей або виключити шкоду машинам або матеріалам, що завдаються, в результаті роздавлювання або розрізу (рис. 3.18).

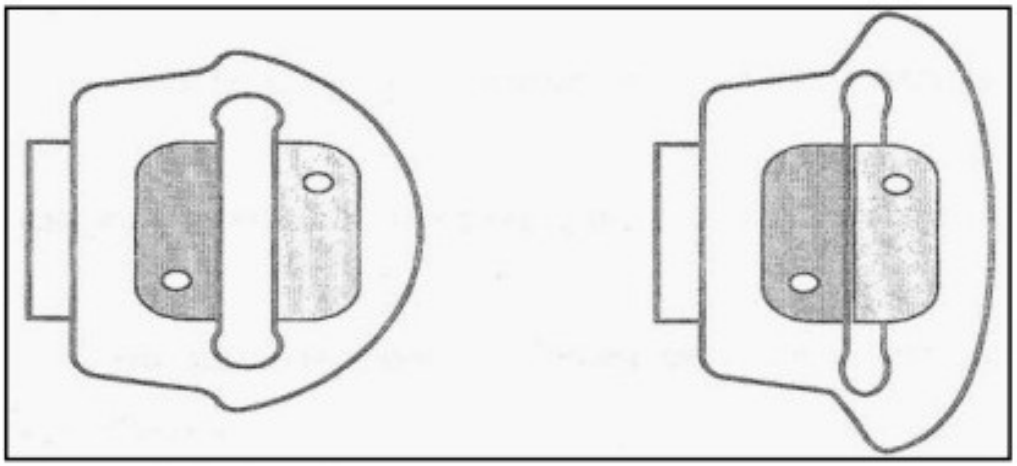

Рисунок 3.18 – Чутливі поверхні

Інтерактивне візуальне програмно-навчальне середовище для робота Robotino View за допомогою блок-схем функції «дистанція», що дозволяє керувати даними датчиками, дає можливість визначати відстань від перешкоди та зупинятися при зіткненні (рис. 3.19) (рис. 3.20) [8].

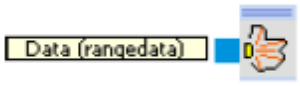

Рисунок 3.19 – Блок-схема функції лазерного далекоміра

Рисунок 3.20 – Блок-схема функції дистанція

Дані лазерного далекоміра відображають дані з додаткового лазерного сканера (рис. 3.21).

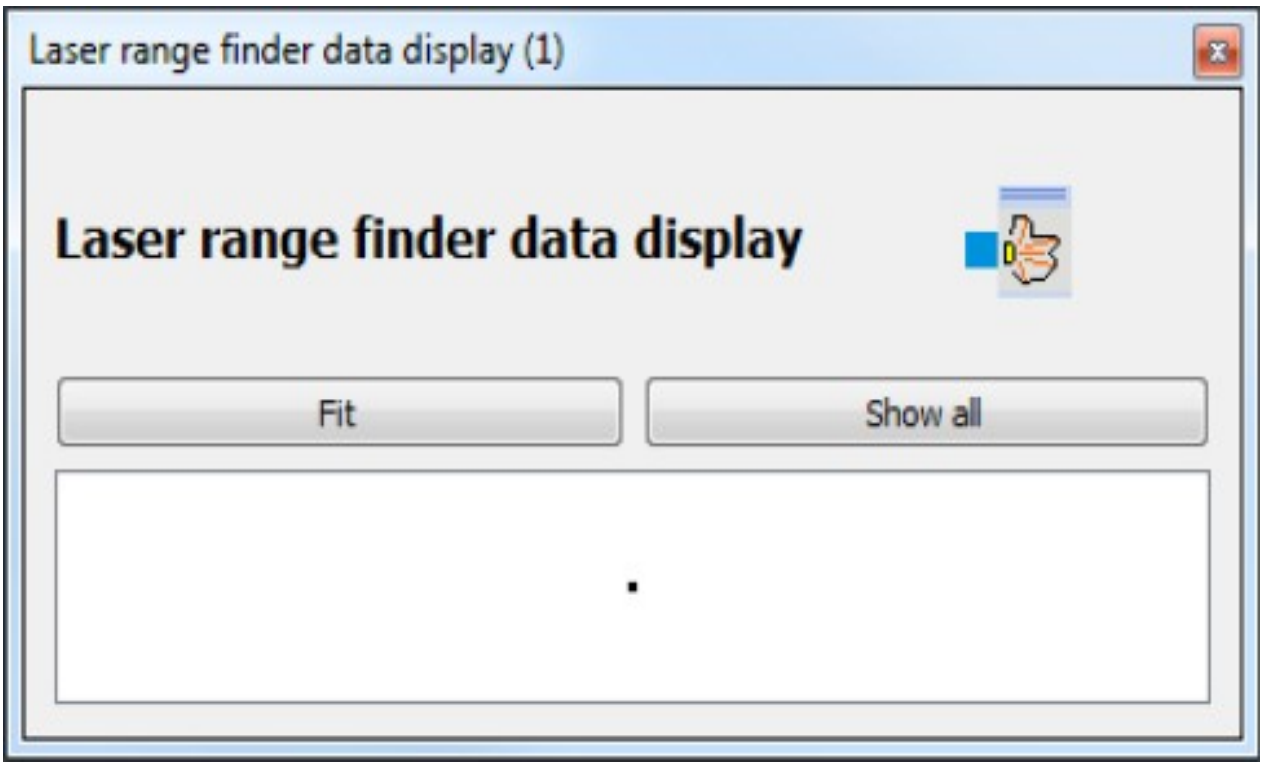

Рисунок 3.21 – Відображення даних лазерного далекоміра

Лазерний далекомір - прилад для вимірювання відстаней із застосуванням лазерного променя.

Широко застосовується в інженерній геодезії, при топографічній зйомці, у військовій справі, у навігації, в астрономічних дослідженнях.

Сучасні лазерні далекоміри в більшості випадків компактні і дозволяють в найкоротші терміни і з великою точністю визначити відстані до об'єктів, що цікавлять.

Лазерні далекоміри розрізняються за принципом впливу на імпульсні та фазові.

3.1.5 Алгоритм JPS

Особливістю управління програмно-аппартним комплексом Robotino полягає в його універсальності.

Його програмувати можна як у візуальній середовищі розробки «Robotino View», так і за допомогою великого кількості бібліотек.

Такі бібліотеки є практично для всіх популярних мов програмування [3]. Компанія «Festo-Didactic» надає продукт для візуальної симуляції роботи робота Robotino.

У вільному доступі є одна стандартна сцена, на якій можна проводити випробування [3].

Особливістю управління програмно-апартним комплексом Robotino полягає в його універсальності.

Його програмувати можна як у візуальній середовищі розробки «Robotino View» так і за допомогою великого кількості бібліотек.

Такі бібліотеки є практично для всіх популярних мов програмування [3]. Компанія «Festo – Didactic» надає продукт для візуальної симуляції роботи робота Robotino.

У вільному доступі є одна стандартна сцена, на якій можна проводити випробування [3].

Етапи проведення досліджень:

– аналіз алгоритму JPS;

– модифікація алгоритму JPS;

– аналіз ПЗ робота 127;

– програмування модифікованого алгоритму.

Розглянемо алгоритм JPS [5]. Даний алгоритм - покращення алгоритму пошуку шляху A *.

Він покращується за рахунок «перескокування» тих позицій, які мають бути переглянуті.

Також слід зазначити, що алгоритм JPS не вимагає додаткової обробки та витрат пам'яті [5]. Розглянемо рішення, яке дає цей алгоритм (рис. 3.20).

Рисунок 3.20 – Розв'язок елементів JPS його модифікація

Шукається оптимальний шлях з точки SP1 до точки DP 1.

Сірим кольором показані стрибкові точки. алгоритм JPS дозволяє знаходити рішення лише для матеріальної точки.

Як очевидно з (рис. 3.20), шукається оптимальний шлях з точки SP1 в точку DP1.

Сірий колір показує стрибкові точки. Алгоритм JPS дозволяє шукати рішення лише матеріальної точки. Для Robotino цей алгоритм не підійде, тому потрібна його модифікація. Вона полягає в тому, щоб уникнути перешкод з урахуванням фізичних розмірів робота без зіткнення з ними.

Модифікація алгоритму JPS полягає в тому, що аналізуються всі діагональні переходи, і якщо зустрічаються перешкоди, шукається альтернативний шлях для даного відрізка.

Результат модифікації зображеннь показано на рис. 3.20.

У візуальному середовищі розробки Robotino view запрограмувати робота можна за допомогою функціональних блоків, які в свою чергу розбиті на категорії. На рис. 3.21 зображено вікно даного середовища розробки.

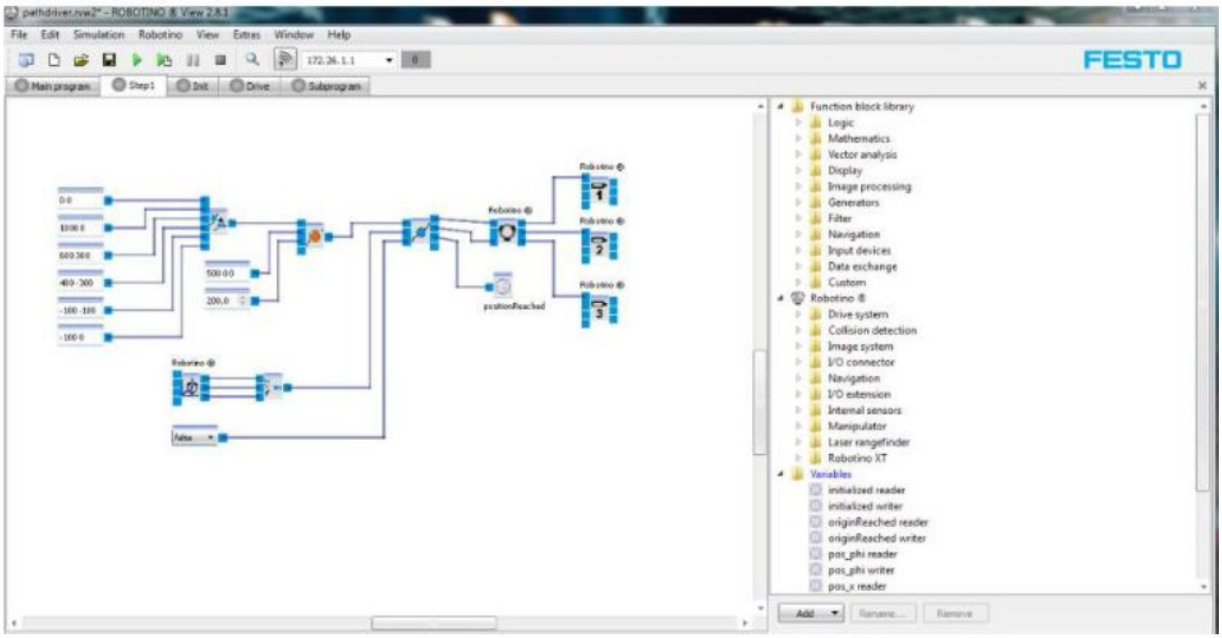

Рисунок 3.21 – Вікно "Robotino view"

На правій панелі є категорії з базовими елементами: Functional block library, Robotino, Variables. У категорії "Functional block library" є набір логічних, математичних операцій, обробка навігації тощо.

Категорія Robotino містить список пристроїв робота, які можна використовувати. Наприклад, у "Drive system" є можливість керувати кожним двигуном окремо. Елементами категорії "Variables" є змінні проекту.

При створенні великого проекту його можна розділити на складові, які розміщують на окремих вкладках.

У центральній сцені користувач формує схему окремої компоненти проекту як послідовність елементів керування.

Під час аналізу програмного забезпечення Robotino було виділено 2 групи програмних засобів: – Robotino view для візуального програмування; – Robotino SIM для моделювання та візуалізації поведінки.

За описом системи програмування Robotino [3] була запропонована ієрархія побудови програмного забезпечення (рис. 3.22):

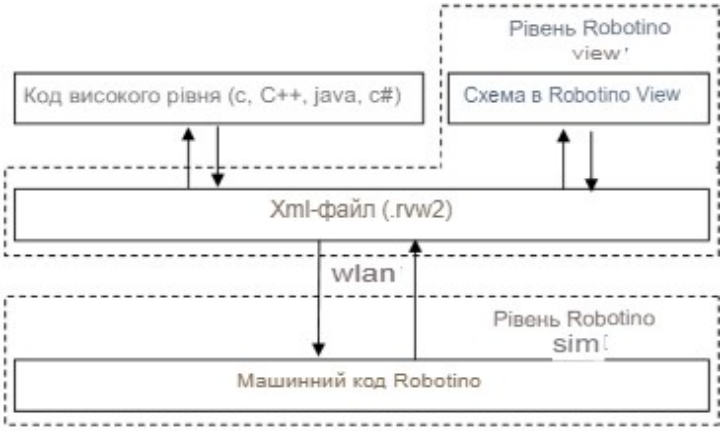

Рисунок 3.22 – Ієрархія побудови ПЗ для Robotino

У цій схемі є два рівні: рівень Robotino view - рівень побудови схеми для управління роботом, і рівень Robotino SIM - рівень представлення даних. Взаємодія між машинним кодом робота та кодом високого рівня здійснюється за допомогою .rvw2-файлу. Цей файл є верстокою xml, причому кожен елемент має свою позицію у файлі. Керування роботом здійснюється дистанційно по бездротовому каналу W LAN (взаємодія між рівнем xml-файлу і машинного коду Robotino). Слід також зазначити, що дані з робота (наприклад дані з датчиків) передаються тим же каналом [3]. У цьому роботі програмування здійснювалося з допомогою мови C#. Компанія-виробник "Festo-Didactic" надає засоби для програмування Robotino у вигляді бібліотеки [6]. Програмування модифікованого алгоритму виконувалось у Microsoft Visual Studio 2008 засобами мови C#. Тестування проводилися в оболонці Robotino SIM, вона дозволяє симулювати переміщення робота. Оцінка даного алгоритму та його модифікації виглядає так (див. табл. 3.1):

Таблиця 3.1 – Оцінки алгоритму (N, M - розмірність карти, K - кількість кроків)

Оцінка алгоритму JPS	Оцінка алгоритму JPS с модифікацією
$F(N,M,K)=(4N+2M)^K$	$F(N,M,K)=(4N+2M)^K+K-1$

Було проаналізовано ієрархію побудови програмного забезпечення для робота Robotino. Також було розроблено програму та реалізовано алгоритм

JPS з модифікацією, що дозволило його застосувати для робота. Виконано оцінку складності даних алгоритмів

3.1.6 Аналіз головного алгоритму

Мобільний робот знаходиться на стартовій точці на тестовій сцені (рис. 3.23) з заздалегідь прокладеними лініями чорного кольору.

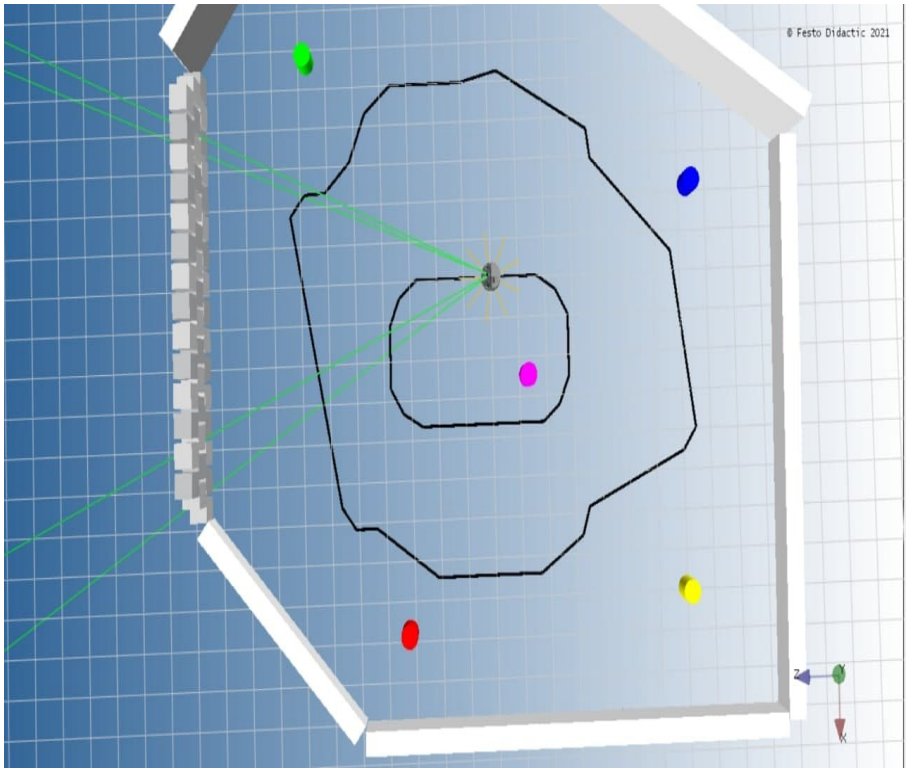

Рисунок 3.23 – Робочій простір мобільного робота

Під час початку роботи використовується алгоритм програмування робота за допомогою блоків одометрії .(рис. 3.23) для початку руху мобільного робота по заздалегідь запрограмованих оператором координатах або по лінії за рахунок СКЗ (Система комп'ютерного зору) як тільки в зону видимості потрапляє обьєкт потрібного нам кольору (рожевого) вмикається функція управління мобільним роботом для знаходження ним заданих кольорів у просторі середовища Robotino View (рис. 3.11).

Завдяки цьому робот починає свій рух по самому короткому маршруту до об’єкта відповідного кольору та зупиняється на заданій оператором відстані (рис. 3.24).

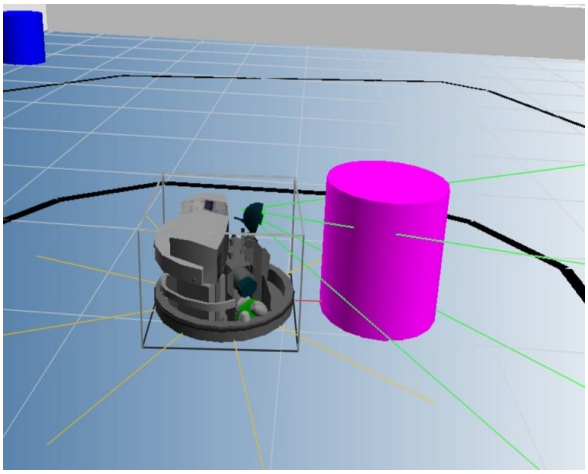

Рисунок 3.24 – Приклад визначення відстані до перешкоди

Відстань задана оператором складає 15 см це реалізовано за допомогою інфракрасного датчика Robotino (рис. 3.17) та блок-схемі функції дистанції (рис. 3.19) за результатом аналізу саме така відстань є оптимальною для правильної роботи зхвату маніпулятора,

Під час наступного єтапу за допомогою Lua скріпту robotino передає інформацію що до місцезнаходження об’єкту який потрібно перемістити.

Приклади макетів зображені на рис. 3.25 та рис. 3.26.

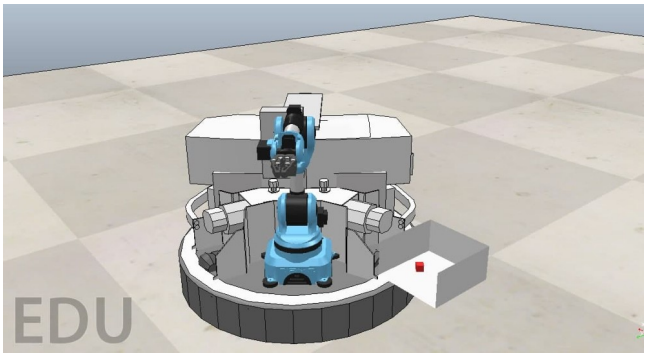

Рисунок 3.25 – Макет роботіно з маніпулятором

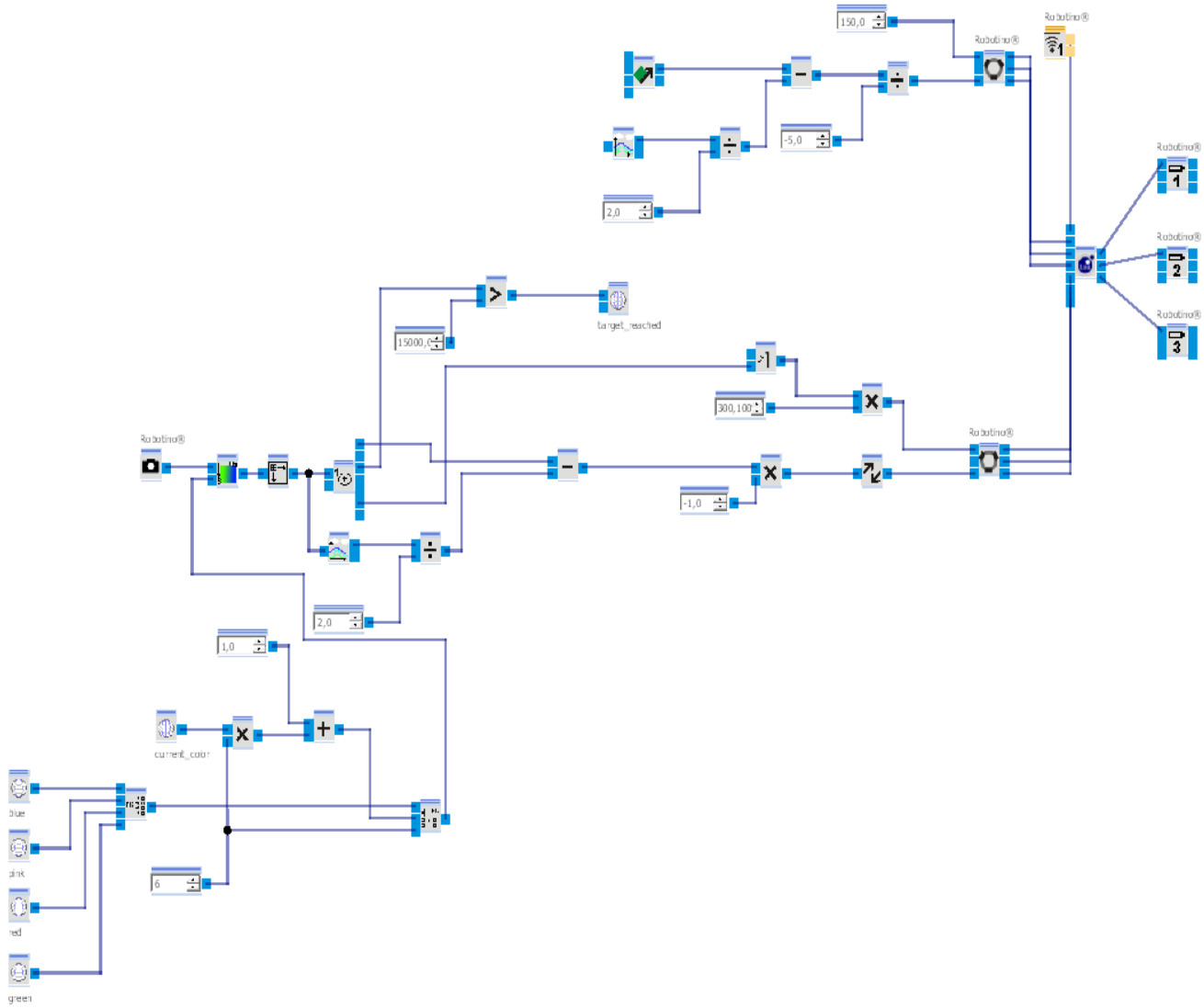

Рисунок 3.26 – Структура головної програми

Одразу після того як маніпулятор закінчив свою роботу robotino прямує до пункту розвантаження вантажу в якому знаходиться платформа певного кольору (синього)на яку за допомогою маніпулятора відбувається розвантаження вантажу.

Також було проаналізовано та розроблено можливість керування мобільним роботом Robotino за допомогою джойстика (рис. 3.1).

Таблиця 3.2 – Порядок виконання алгоритмів головної програми

№	Назва алгоритму	Опис	Час виконання
1	Алгоритм руху по координатам з використанням блоків одометрії	Мобільний робот рухається по заздалегідь заданим координатам	30 с
2	Алгоритм пошуку кольору сегмента	Рух до об'єкта потрібного кольору а саме рожевого на відстань задану оператором (15 см)	20 с
3	Очікування	Робот чекає завершення роботи маніпулятора по завантаженню об'єкта	25 с
4	Алгоритм руху по координатам з використанням блоків одометрії	Мобільний робот рухається по заздалегідь заданим координатам до зони відвантаження	30 с
5	Алгоритм пошуку кольору сегмента	Рух до об'єкта потрібного кольору а саме синього на відстань задану оператором (15 см)	20 с
6	Очікування	Робот чекає завершення роботи маніпулятора по відвантаженню об'єкта	25 с

3.2 Висновки до третього розділу

Було розроблено ряд алгоритмів програмування мобільного робота Robotino на платформі RobotinoView .

У ході ознайомлення я зрозумів, що такі робототехнічні системи мають широкий спектр можливостей, а блокове програмування робить процес простим, та зрозумілим .

4 ОХОРОНА ПРАЦІ

Природне освітлення має здійснюватись через світлові прорізи, орієнтовані переважно на північ чи північний схід, і забезпечувати коефіцієнт природної освітленості (КПО) не нижче, ніж 1,5%.

Гігієнічні вимоги до параметрів виробничого середовища включають вимоги до параметрів мікроклімату, освітлення, шуму й вібрації, рівнів електромагнітного та іонізуючого випромінювання. У виробничих приміщеннях на робочих місцях із ПК мають забезпечуватись оптимальні значення параметрів мікроклімату: температури, відносної вологості й рухливості повітря.

Норми мікроклімату для приміщень з ПК: у холодну пору року температура повітря при категорії робіт легка-1а має бути 22-24 ºС, вологість 40-60%, при категорії робіт легка-1б температура має бути 21-23 ºС, вологість також 40-60 %; у теплу пору року при категорії робіт а температура в приміщеннях має бути 23-25 ºС з вологістю 40-60%, а при категорії робіт 1б температура має бути 22-24 ºС з такою самою вологістю.

Значення освітленості на поверхні робочого столу в зоні розміщення документів має становити 300–500 лк, Якщо це неможливо забезпечити системою загального освітлення, допускається використовувати місцеве освітлення.

Конструкція робочого місця користувача ПК має забезпечити підтримання оптимальної робочої пози.

Робочі місця із ПК слід так розташовувати відносно світлових прорізів, щоб природне світло падало збоку, переважно зліва.

При розміщенні робочих столів із персональним комп’ютером слід дотримуватись таких відстаней: між бічними поверхнями – 1,2 м; від тильної поверхні одного ПК до екрана іншого – 2,5 м.

Екран ПК має розташовуватися на оптимальній відстані від очей користувача, що становить 600…700 мм.

Клавіатуру слід розташовувати на поверхні столу на відстані 100…300 мм від краю, звернутого до працюючого.

Вогонь, що вийшов із-під контролю, здатний викликати значні руйнівні та смертоносні наслідки. До таких проявів вогняної стихії належать пожежі.

Пожежа – неконтрольоване горіння поза спеціальним вогнищем, що розповсюджується у часі і просторі.

Основними напрямками забезпечення пожежної безпеки є усунення умов виникнення пожежі та мінімізація її наслідків.

Об'єкти повинні мати системи пожежної безпеки, спрямовані на запобігання пожежі, дії на людей та матеріальні цінності небезпечних факторів пожежі, в тому числі їх вторинних проявів.

До таких факторів належать: полум'я та іскри, підвищена температура навколишнього середовища, токсичні продукти горіння й термічного розкладу матеріалів і речовин, дим, знижена концентрація кисню.

Вторинними проявами небезпечних факторів пожежі вважаються: уламки, частини зруйнованих апаратів, агрегатів, установок, конструкцій; радіоактивні та токсичні речовини і матеріали, викинуті зі зруйнованих апаратів та установок; електричний струм, пов'язаний з переходом напруги на струмопровідні елементи будівельних конструкцій, апаратів, агрегатів внаслідок пошкодження ізоляції під дією високих температур; небезпечні фактори вибухів, пов'язаних з пожежами; вогнегасні речовини.

Системи пожежної безпеки – це комплекс організаційних заходів і технічних засобів, спрямованих на запобігання пожежі та збитків від неї.

Пожежна безпека об'єкта повинна забезпечуватися системою запобігання пожежі, системою протипожежного захисту і системою організаційно-технічних заходів.

Системи пожежної безпеки мають запобігти виникненню пожежі і впливу на людей небезпечних факторів пожежі на необхідному рівні. Потрібний рівень пожежної безпеки людей за допомогою зазначених систем не повинен бути меншим за 0,999999 відвернення впливу на кожну людину, а

допустимий рівень пожежної небезпеки для людей не може перевищувати 10" впливу небезпечних факторів пожежі, що перевищують гранично допустимі значення на рік у розрахунку на кожну людину [25].

Об'єкти, пожежі на яких можуть призвести до загибелі або масового ураження людей небезпечними факторами пожежі та їх вторинними проявами, а також до значного пошкодження матеріальних цінностей, повинні мати системи пожежної безпеки, що забезпечують мінімально можливу імовірність виникнення пожежі. Конкретні значення такої імовірності визначаються проектувальниками та технологами.

Метою пожежної безпеки об'єкта є попередження виникнення пожежі на визначеному чинними нормативами рівні, а у випадку виникнення пожежі – обмеження її розповсюдження, своєчасне виявлення, гасіння пожежі, захист людей і матеріальних цінностей.

Основними вихідними даними при розробці комплексу технічних і організаційних рішень щодо забезпечення потрібного рівня пожежної безпеки в кожному конкретному випадку є чинна законодавча і нормативно-технічна база з питань пожежної безпеки, вибухо- і пожежонебезпечні властивості матеріалів і речовин, що застосовуються у виробничому циклі, кількість вибухо- і пожежонебезпечних матеріалів і речовин і особливості виробництва.

Саме залежно від категорії приміщень та будівель і класу зон за вибуховою і пожежною небезпекою, відповідно до вимог чинних нормативів, розробляються технічні і організаційні заходи і засоби забезпечення вибухопожежної безпеки об'єкта.

4.1 Висновки до четвертого розділу

Охорона праці у наш час є важливою складовою у роботі програміста.

Відділи охорони праці забезпечують комфортні та безпечні умови праці для програмістів.

ВИСНОВКИ

У процесі виконання магістриської роботи було розроблене програмне забезпечення, за допомогою якого виконується рух мобільного транспортного робота Robotino по робочій зоні приміщення та алгоритм пошуку обьєктів по кольору. В процесі розробки був задіяний метод комп'ютерного моделювання для подальшого тестування програмних компонентів мобільного робота. Було проведено тестування – за допомогою програмного забезпечення для симуляції роботів

Розробка ПЗ відбувалася у середовищі програмування Robotino View із застосуванням Robotino SIM для створення комп'терної моделі приміщення для тестування Festo Robotino.

Метою роботи була розробка удосконаленого програмного забезпечення для маніпуляційної системи мобільного робота

Розвиток даного проєкту в майбутньому має великий простір для мондернізації та допрацювання створена система дозволить тестувати подальші розробки, не докладаючи великих зусиль для перенесення моделі у реальний світ. Програма написана в окремих функціях, що дозволяє легко додавати або змінювати функціонал системи для тестування конкретних елементів моделі або всієї моделі в цілому .

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008:2015. Документація. Звіти у сфері науки та техніки. Структура та правила оформлення. [Чинний від 2015–06–22]. Вид. офіц. Київ, 2016. 29 с. (Інформація та документація).

2. Дипломне проектування для студентів усіх форм навчання спеціальностей 151 «Автоматизація та комп'ютерно-інтегровані технології»: / упоряд. І.Ш. Невлюдов, А.О. Андрусевич, О.В. Токарєва, Г.В. Пономарьова. Київ, 2018. 320 с.

3. Методичні вказівки з підготовки й оформлення кваліфікаційної роботи здобувачами другого (магістерського) рівня вищої освіти спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології, освітньо-професійних програм: «Автоматизоване управління технологічними процесами», «Комп'ютерно-інтегровані технологічні процеси і виробництва», «Комп'ютеризовані та робототехнічні системи» / упоряд. І. Ш. Невлюдов, Р. В. Артюх, Н. П. Демська, В. В. Євсєєв, О. І. Филипенко, О. М. Цимбал. Харків : ХНУРЕ, 2021. 50 с.

4. Цимбал О.М., Бронніков А.І. Системи адаптації роботів і технологія OpenCV: Навчальний посібник – Харків: ХНУРЕ, 2019. – 148 с.

5. I. Nevliudov, O. Tsymbal, A. Andrusevitch, V. Gopejenko. Intelligent Decision-Making Support for Flexible Integrated manufacturing – Riga: ISMA, 2020. – 390 p.

6. Nevlyudov I., Tsymbal O., Bronnikov A. Intelligent means in the system of managing a manufacturing agent / Сучасний стан наукових досліджень та технологій в промисловості. 2018. № 1 (3). – С. 33-47.

7. В.М. Шарапов, Е.С. Полищук, Н.Д. Кошевой, Г.Г. Ишанин, И.Г. Минаев, А.С. Совлуков // Датчики: Справочное пособие / Москва: Техносфера, 2017. 624 с. https://www.bytemag.ru/articles/detail.php?ID =6470. (дата звернення 15.11.2019).

8. C. A. Grimes, E. C. Dickey, and M. V. Pishko (2018), Encyclopedia of Sensors (10-Volume Set), American Scientific Publishers. P. 503-507.

9. Кремлев А.С., Зимеенко К.А., Боргуль А.С. Моделювання та програмування робото технічних комплексів // НИУ ИТМО СПб,2017. С. 136. http://www.festodidactic.com/int-en/learning-systems/education-and-research-robots-robotino (дата звернення 09.11.21).

10. Дослідження моделі групової організації автономних роботів в умовах не детермінованого робочого пр остору. 2018: зб. / 19 международный молодежный форум «Радиоэлектроника и молодеж в 21 веке». Харків, 2018. С. 79-80.

11. Розробка програмних засобів для візуального керування мобільним роботом. 2016: зб. / 20 международный молодежный форум «Радиоэлектроника и молодеж в 21 веке». Харків, 2018. С. 9-10.

12. Цололо С.О., Бровкіна Д.Ю., Кравецький А.О. Розробка системи керування мобільним роботом з можливістю виявлення перешкод на основі відеоданих // Наукові праці ДонНТУ 2017/ №1(20). С 93-98.

13. Методи розпізнавання образів на основі системи комп'ютерного зору. 2018: зб. / 19 международный молодежный форум «Радиоэлектроника и молодеж в 21 веке». Харків, 2018. С. 19-20.

14. Програмний модуль для ідентифікації подальшої траєкторії перемішення об'єкту. 2018: зб. / 20 международный молодежный форум «Радиоэлектроника и молодеж в 21 веке». Харків, 2018. С. 55-56.

15. Використання систем адаптивного керування для автоматизації промислових процесів логістики. 2018: зб. / 20 международный молодежный форум «Радиоэлектроника и молодеж в 21 веке». Харків, 2018. С. 123-124.

16. Education and Research Robots: Robotino // Festo Didactic. 2017. http://www.festodidactic.com/int-en/learning-systems/education-and-research-robots-robotino (дата звернення 09.11.21).

17. Robotino View // Festo Didactic. 2016. http://www.festo- didactic.com/ int-en/learningsystems/software-e-learning/robotino-sim-view/robotino-view.htm?

fbid=aW50LmVuLjU1Ny4xNy4 xOC4xMjE5LjcyNjY (дата звернення 09.11.21).

18. Robotino Sim// Festo Didactic. 2017 : https://www.festodidactic.com/intn/services/robotino/simulation/?fbid= aW50LmVuLjU1Ny4xNy4zNC4xNDQy (дата звернення 15.12.2021).

19. Microsoft Robotics Studio – robotehnika dlya vseh [Microsoft Robotics Studio – robotics for everyone]. https://www.bytemag.ru/articles/detail.php?ID=6470. (дата звернення 15.11.2021).

20. Lektsiya 7: Raspredelennoe immitatsionnoe modelirovanie [Lecture 7: distributed simulation modeling]. Natsionalnyi Otkrytyi Universitet - INTUIT‖ [National Open University - INTUIT‖]. Available to: http://www.intuit.ru/studies/courses/1146/238/lecture/6150?page=1&keyword_content=discrete%20event%20simulation. (дата звернення 19.11.2021).

21. Ivanova S.B, Salnicov I. S., Salnicov R. I. Multipurpose robotic computing hardware systems: needs and problems of development. Problemy iskusstvennogo intellekta [Problems of Artificial Intelligence], 2019, no. 0 (1), pp. 50-60.

22. Robotics. 3D CAD Design Software | SOLIDWORKS. Available at: http://www.solidworks.com/sw/education/robotic-design-resources-students.htm. (accessed 10.11.2021).

23. NXT-G: the development environment supplied with Lego Mindstorms, NXT-G. Génération Robots. Available at: http://www.generationrobots.com/blog/en/2015/09/nxt-g-the-development-environment-supplied-with-lego-mindstorms-nxt-g. (accessed 10.12.2021).

24. TRIK Studio, Всё о ТРИК: TRIK Studio [Everything about TRIK: TRIK Studio]. Available at: http://blog.trikset.com/p/trik-studio.html. (accessed 14.11.2021).

25. Robotics System Toolbox. MathWorks - Makers of MATLAB and Simulink. MATLAB & Simulink. Available at: https://www.mathworks.com/products/robotics. (accessed 22.11.2021).

26. anyKode Marilou. anyKode Marilou - Modeling and simulation environment for Robotics. Available at: http://doc.anykode.com/frames.html?frmname=topic&frmfile=index.html. (accessed 21.11.2021).

27. What is Arduino? Arduino. Available at: https://www.arduino.cc/en/Guide/Introduction. (accessed 14.12.2021).