ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра комп'ютерних наук

Пояснювальна записка

до бакалаврської роботи
на ступінь вищої освіти бакалавр

на тему: «Застосування штучного інтелекту для керування безпілотним транспортним засобом в умовах промислового майданчика»

Виконав: студент 4 курсу, групи КНД–41
спеціальності 122 Комп'ютерні науки
Попов А. О.
(прізвище та ініціали)

Керівник Звенигородський О. С.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

Нормоконтроль Горєшнєва Н.А.
(прізвище та ініціали)

Київ – 2021

ВСТУП

Штучний інтелект та нейронні мережі в нашому світі, що постійно розвивається набувають дедалі більшої популярності: розпізнавання обличь для обробки фото, віртуальна реальність та ігрова індустрія, автоматизація та управління транспортом. Все тому, що в світі перевантаженому подіями та інформацією старий підхід до розробки систем стає мало ефективним, і якщо компанії хочуть продовжувати прогресивно розвиватись і подивувати свої прибутки їм потрібно адаптуватись до сучасних потреб. Тим більше що системи під керуванням штучного інтелекту стають дедалі ефективнішими, дозволяють за короткий проміжок часу виконувати величезну кількість роботи, що звичайно позитивно впливає на доходи компанії. При використанні застарілих систем, які при виконанні роботи приймають рішення на основі заздалегідь продуманої програми поведінки, можуть виникати вузькі місця зв’язані перш за все з тим, що коли розробник програмує таку систему він опирається тільки на свій досвід в цій справі, але часто продумати всі варіанти розвитку подій просто не можливо, тоді виникає необхідність в тому, щоб програма сама приймала рішення опираючись на інформацію якою вона володіє в даний момент часу. Тож штучний інтелект - це сукупність системних, програмних та конструктивних рішень в програмуванні та розробці програмного забезпечення і приладів які його використовують, покликаних для ефективного виконання певної корисної роботи, наприклад візуальне сприйняття об’єктів та прийняття рішень. Саме прийняття рішень являє собою основну характеристику штучного інтелекту, адже здатність системи раціоналізувати проблеми і знаходити оптимальний план її вирішення з найбільшою точністю при найменших затратах відділяє його від програм з прописаним сценарієм. Використання таких система в управлінні транспортом на промисловому майданчику могло б суттєво полегшити та пришвидшити роботу працівників підприємств, а значить і принести користь їх власникам.

1 РОЗДІЛ ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

1.1 Огляд безпілотних технологій

Безпілотний транспортний засіб – це транспортний засіб оснащений системами автоматизованого управління без прямої участі людини. Безпілотні транспорті засоби можуть рухатись завдяки спеціальному програмному навчанню і сенсорам. Програмне забезпечення керує роботою усіх систем автомобіля: поворотом руля, перемиканням передач, газом та тормозом. Сенсори збирають інформацію про навколишнє середовище, яка лягаю в основу дій безпілотного автомобіля. Основною частиною такої системи управління зазвичай виступає штучний інтелект, використовуються технології розпізнавання образів та нейронні мережі. Деякі системи покладаються на інфраструктурні рішення наприклад, вбудовані в покриття дороги або біля неї, але більш продвинуті технології здатні рухатись в тих же умовах, що і людина приймаючи рішення по зміні швидкості руху, перемиканні передачі та зміні куту повороту руля на основі даних які надходять з сенсорів системи.

Всього існує шість рівнів автопілоту:

0. рівень, в транспорті абсолютно немає ніяких систем які допомагають керманичу управляти авто
1. рівень, машина управляється керманичом та можуть працювати системи круїз-контролю, паркування та системи і системи попередження сходу з полоси
2. 2 рівень – часткова автоматизація, система повністю керує автомобілем, прискорення, сповільнення управління рулем, водій сидить за рулем та готовий втрутитись в будь який момент часу коли система не зможе нормально відреагувати.

3. 3 рівень – умовна автоматизація, від водія не потребується минева реакція, він може займатись своїми справами сидячи на сидінні за кермом. Система сама реагує на ситуації потребуючі миттєвих дій, таких як гальмування.
4. 4 рівень – широка автоматизація, відрізняється від 3 рівня тим, що водію не потрібно постійно слідкувати за процесом, він може навіть піти спати. В ситуації коли система не може вирішити проблему вона сама зупиниться і перейде в режим очікування в безпечному місці.
5. 5 рівень – повний автоматизація, непотрібно ніякого людського впливу на систему.

Для того щоб автомобіль зміг безпечно рухатись йому потрібно бачити, думати та мати карту дороги по який він рухаються. Автомобілі з автопілотом мають вбудовані системи навігації та датчики, які дозволяють йому самостійно рухатись по дорозі та розпитувати чіткий маршрут. Для того щоб навчити автомобіль бачити потрібно використовувати системи комп’ютерного бачення. Для того щоб навчити автомобіль думати потрібно використовувати системи на основі штучного інтелекту. Для того щоб дати авто карту її потрібно створити і навчити працювати з нею.

Для створення безпілотних систем які виконують певну корисну роботу, наприклад перевезення вантажів чи роботу по переміщенню матеріалів на будівництві потрібно спочатку визначити в яких умовах буде працювати система. Визначити з якими типами вантажів якими методами вона буде користуватись для їх переміщення

1.2 Характеристики штучного інтелекту та машинного навчання

Ідеальною характеристикою штучного інтелекту являється його здатність раціоналізувати і вживати дії, які мають найбільші шанси на досягнення конкретної цілі. Підмножиною штучного інтелекту є машинне навчання, яке

стосується концепції, згідно з якою комп’ютерні програми можуть автоматично вчитися на нових даних і пристосовуватися до них без допомоги людей. Методи глибокого навчання дозволяють це автоматичне навчання шляхом поглинання величезної кількості неструктурованих даних, таких як текст, зображення чи відео. Штучний інтелект базується на принципі, згідно з яким людський інтелект можна визначити таким чином, щоб машина могла легко його імітувати та виконувати завдання, починаючи від найпростіших і закінчуючи ще більш складними. Цілі штучного інтелекту включають імітацію пізнавальної діяльності людини. Дослідники та розробники в галузі роблять напрочуд швидкими кроками імітацію таких видів діяльності, як навчання, міркування та сприйняття, настільки, наскільки це можна конкретно визначити. Деякі вважають, що новатори, можливо, незабаром зможуть розробити системи, які перевершують здатність людей вчитися чи обґрунтовувати будь-яку тему. Але інші залишаються скептично налаштованими, оскільки вся пізнавальна діяльність пов’язана з ціннісними судженнями, які підпорядковуються людському досвіду.

Машинне навчання - це галузь штучного інтелекту (ШІ) та інформатики, яка зосереджена на використанні даних та алгоритмів для імітації способу навчання людей, поступово покращуючи його точність.

Воно є важливою складовою зростаючої галузі науки про дані. За допомогою статистичних методів алгоритми навчаються робити класифікації або прогнозування, розкриваючи ключові уявлення в рамках проектів з видобутку даних. Ці ідеї згодом стимулюють прийняття рішень у програмах та бізнесі, в ідеалі впливаючи на ключові показники зростання. У міру того, як великі дані продовжуватимуть розширюватися та зростати, ринковий попит на дослідників даних буде зростати, вимагаючи від них допомоги у визначенні найбільш актуальних бізнес-питань, а згодом даних для відповіді на них.

Класифікатори машинного навчання поділяються на три основні категорії:

Контрольоване машинне навчання

Контрольоване навчання, також відоме як кероване машинне навчання, визначається використанням маркованих наборів даних для навчання алгоритмів, які дозволяють класифікувати дані або точно прогнозувати результати. Коли вхідні дані подаються в модель, вона регулює свою вагу, поки модель не буде встановлена належним чином. Це відбувається як частина процесу перехресної перевірки, щоб гарантувати, що модель дозволяє уникнути переобладнання або недооснащення. Навчання під контролем допомагає організаціям вирішувати різноманітні масштабні проблеми, наприклад класифікувати спам в окремій папці з папки "Вхідні". Деякі методи, що використовуються в контрольованому навчанні, включають нейронні мережі, наївні баєси, лінійну регресію, логістичну регресію, випадковий ліс, машину з підтримкою векторів (SVM) тощо.

Машинне навчання без нагляду

Навчання без нагляду, також відоме як машинне навчання без нагляду, використовує алгоритми машинного навчання для аналізу та кластеризації немічених наборів даних. Ці алгоритми виявляють приховані шаблони або групування даних без потреби втручання людини. Його здатність виявляти подібності та відмінності в інформації роблять це ідеальним рішенням для дослідницького аналізу даних, стратегій перехресних продажів, сегментації клієнтів, розпізнавання зображень та зразків. Він також використовується для зменшення кількості об’єктів у моделі за допомогою процесу зменшення розмірності; Аналіз основних компонентів (PCA) та декомпозиція особливих значень (SVD) - два загальних підходи для цього. Інші алгоритми, що використовуються при неконтрольованому навчанні, включають нейронні мережі, k-означає кластеризацію, імовірнісні методи кластеризації та багато іншого.
Навчання під наглядом

Навчання під наглядом

Навчання під наглядом пропонує щасливе середовище між навчанням під наглядом та без нагляду. Під час навчання він використовує менший набір

маркованих даних для керівництва класифікацією та вилучення ознак із більшого, немаркованого набору даних. Навчання під наглядом може вирішити проблему недостатньої кількості маркованих даних (або неможливості дозволити собі маркування достатньої кількості даних) для підготовки керованого алгоритму навчання.

Нейронна мережа, комп’ютерна програма, яка працює таким чином, що надихається природною нейронною мережею мозку. Завданням таких штучних нейронних мереж є виконання таких когнітивних функцій, як вирішення проблем та машинне навчання. Основною привабливістю нейронних мереж є їх здатність наслідувати навички розпізнавання образів мозку. Серед комерційних застосувань цієї здатності нейронні мережі використовувались для прийняття інвестиційних рішень, розпізнавання почерку та навіть виявлення бомб.

Відмінною рисою нейронних мереж є те, що знання про її домен поширюються по всій мережі, а не явно записуються в програму. Ці знання моделюються як зв’язки між оброблювальними елементами (штучними нейронами) та адаптивними вагами кожного з цих зв’язків. Потім мережа вчиться, потрапляючи в різні ситуації. Нейронні мережі здатні досягти цього, регулюючи вагу з'єднань між зв'язуючими нейронами, згрупованими в шари, як показано на малюнку простої мережі прямої передачі. Вхідний шар штучних нейронів отримує інформацію з навколишнього середовища, а вихідний рівень передає відповідь; між цими шарами може бути один або кілька «прихованих» шарів (без прямого контакту з навколишнім середовищем), де відбувається більша частина обробки інформації. Вихід нейронної мережі залежить від ваги зв’язків між нейронами в різних шарах. Кожна вага вказує на відносну важливість конкретного з'єднання. Якщо сукупність усіх зважених входів, отриманих певним нейроном, перевищує певне порогове значення, нейрон надішле сигнал кожному нейрону, до якого він підключений, у наступному шарі. Наприклад, під час обробки заявок на

позику вхідні дані можуть представляти дані профілю заявника на позику та вихідні дані щодо надання позики.

Дві модифікації цієї простої прямої нейронної мережі сприяють зростанню додатків, таких як розпізнавання обличчя. По-перше, мережа може бути оснащена механізмом зворотного зв'язку, відомим як алгоритм зворотного розповсюдження, який дозволяє їй регулювати ваги з'єднання назад через мережу, навчаючи її у відповідь на репрезентативні приклади. По-друге, можуть бути розроблені повторювані нейронні мережі, що включають сигнали, що проходять в обох напрямках, а також всередині та між шарами, і ці мережі здатні до значно складніших моделей асоціації. (Насправді для великих мереж може бути надзвичайно складно точно простежити, як визначався вихідний результат.)

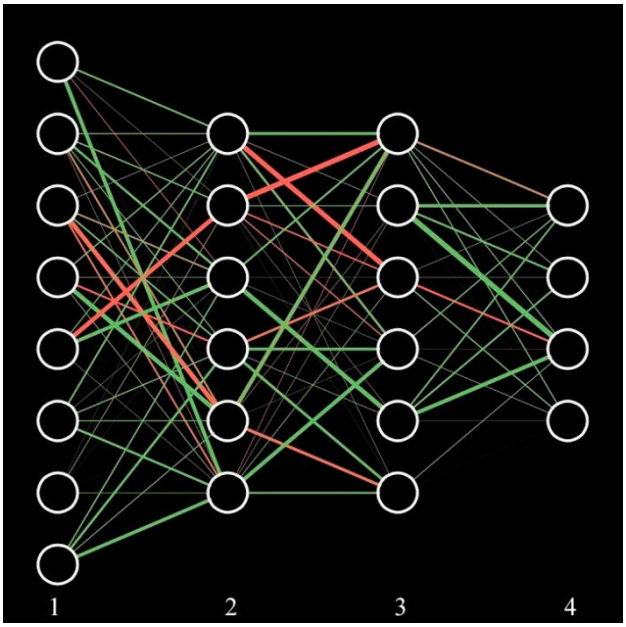

Рисунок 1.1. Нейронна мережа

Рівні нейронної мережі

1. Вхідний шар
2. Прихований шар
3. Прихований шар
4. Вихідний шар

1.3 Систем розпізнавання образів

Розпізнавання образів – відноситься до однієї з найважливіших задач пов’язаних з штучним інтелектом. Так як саме завдяки розпізнаванню образів можлива реалізація систем автопілоту. До розпізнавання образів в автоматизації відносяться різні задачі, такі як розпізнавання цілей і об’єктів, навігація, захоплення і маніпуляція, взаємодія людини і машини. Здешевлення технології створення камер високої роздільної здатності, зменшення їх габаритних розмірів та збільшення комплексу функціональності, робить їх найкращим вибором в якості сенсорів для робототехніки та автоматизації.

Рис. 1.2. Розпізнані образи авто на трасі

Ціль комп’ютерного зору – розробка методів, які дозволяють машині розуміти або аналізувати зображення та відео. Найважливішим моментом при розробці модулю технічного зору – методи, які використовуються в ньому. Якщо

ми говоримо про розпізнавання візуальних образів, то це найважливіша задача комп’ютерного зору.

Машинний зір представляє собою промислове застосування комп’ютерного зору та його конкретні реалізації.

Обробка зображень дає дослідникам набір методів для конвертування зображень в цифрову форму і виконання різних операцій над ними для отримання додаткової інформації.

Машинне бачення – це системи на основі комп’ютерного бачення які використовуються на виробництвах і в промисловій автоматизації. В той час як звичайне комп’ютерне бачення являє собою загальний набор методів для забезпечення комп’ютерів зором, областю інтересів машинного зору, як направлення в інженерії, являються цифрові пристрої вводу-виводу і комп’ютерні мережі, створені для контролю над продукцією яку випускає завод, контролю виробничого обладнання та відбракування неякісно виконаних деталей. Машинне бачення являє собою підрозділ інженерії, зв’язаний з обчислюваною технікою, оптикою, машинобудуванням та промисловою автоматизацією. Такі системи використовуються в легкій та важкій промисловості для виконання рутинних і трудомістких задач, відбракування плат на виробництві, зборка систем на основі плат куди потрібно впаювати мікро чіпи.

При фільтруванні на зображенні можна виділити області, на такому етапі аналізування зображення не проводиться, але точки які пройшли фільтрацію, можна розглянути як точки з особливими характеристиками.

Навчання нейронних мереж зазвичай включає навчання під контролем, де кожен приклад навчання містить значення як вхідних даних, так і бажаного результату. Як тільки мережа зможе виконати достатньо добре додаткові тести, вона може бути застосована до нових випадків. Наприклад, дослідники з Університету Британської Колумбії підготували нейронну мережу, що передає

дані, з даними про температуру та тиск з тропічного Тихого океану та з Північної Америки для прогнозування майбутніх глобальних моделей погоди.

На відміну від цього, певні нейронні мережі навчаються за допомогою неконтрольованого навчання, в якому мережа представлена колекцією вхідних даних і дана мета виявлення закономірностей - без пояснення того, що конкретно шукати. Така нейронна мережа може бути використана в аналізі даних, наприклад, для виявлення кластерів клієнтів у сховищі маркетингових даних. Нейронні мережі є на передньому краї когнітивних обчислень, які призначені для того, щоб інформаційні технології виконували деякі з найдосконаліших психічних функцій людини. Системи глибокого навчання базуються на багатошарових нейронних мережах та потужності, наприклад, можливості розпізнавання мови мобільного помічника Apple від компанії Siri. У поєднанні з експоненціально зростаючою обчислювальною потужністю та масивними сукупностями великих даних нейронні мережі глибокого навчання впливають на розподіл роботи між людьми та машинами.

Object Tracking - дуже цікавий напрямок, яке вивчається і еволюціонує не перший десяток років. Зараз багато розробки в цій області побудовані на глибокому навчанні, яке має перевагу над стандартними алгоритмами, так як нейронні мережі можуть апроксимувати функції часто краще. Таким чином розпізнавання образів на даний момент є найефективнішим напрямком для використання неорних мереж, оскільки нейронні мережі імітують процес мислення людини і можуть хоч і не зі сто процентною точністю, але розпізнати об’єкти на зображенні. А при будуванні авто, що будуть керуватись ШІ така функція як розпізнавання образів є однією з найважливіших [1].

1.4 Огляд існуючих технологій безпілотного керування авто

Так в компанії Tesla використовується система з використанням великої кількості оптичних камер, машина по справжньому бачить дорогу і будую карту місцевості. За допомогою трьох фронтальних камер, однієї з широким кутом огляду, так називаним риб'ячим оком, яка допомагає бачити периферійні об'єкти, об'єкти спереду машини, розпізнає машини які їдуть в сусідніх полосах, дорожні знаки, сигнали світлофора на відстані до 60 метрів. Камера з звичайним кутом зору яка розпізнає об'єкти на відстані 150 метрів і захоплює максимальний спектр випадків застосування та камера з вузьким полем зору, яка дає сфокусоване зображення віддалених об'єктів на відстані 250 метрів, корисний при русі на великій швидкості. Передні бокові камери розміщені з обох боків авто, з кутом огляду в 90 градусів направлені вперед, фіксують машини які зненацька виїжджають на вашу полосу на шосе, і гарантують безпечність при водінні з обмеженим кутом зору. Бокові камери заднього виду з обох сторін автомобіля, що важливо для безпечної зміни полоси руху. Камера заднього виду яка страхує задні бокові камери і датчики, використовуються для попередження зіткнень ззаду, а також при виконанні поковочних маневрів [2].

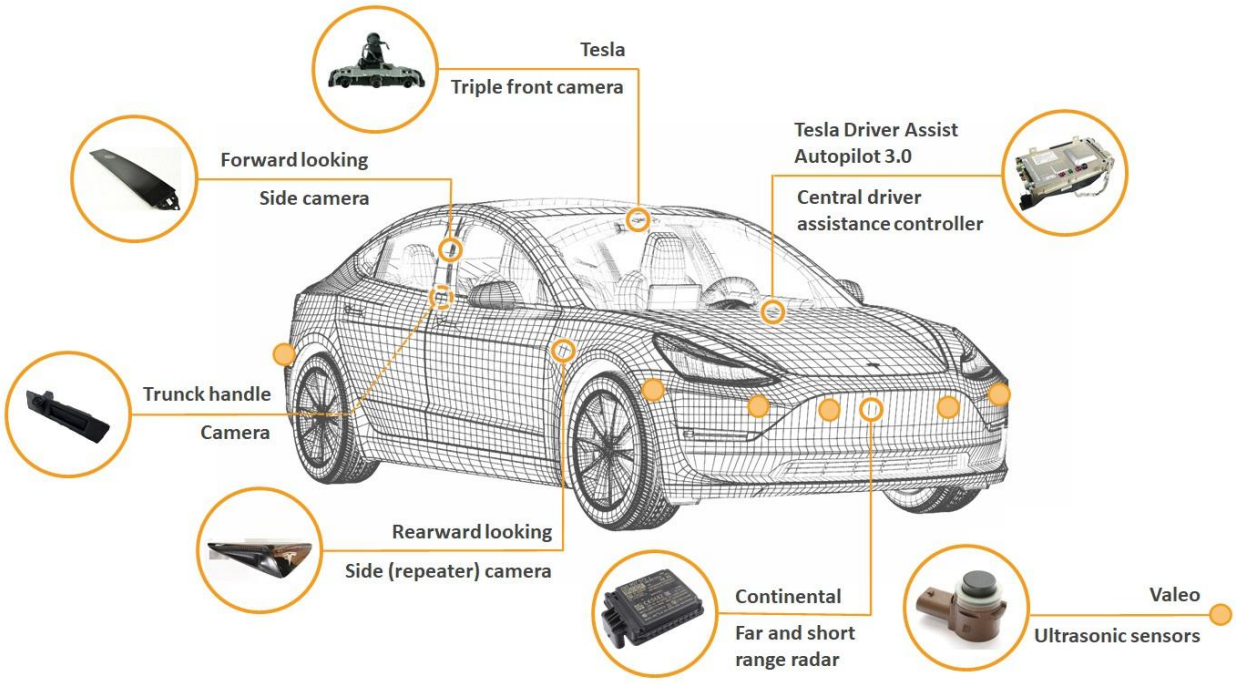

Рис. 1.3. Тесла та всі датчики в ній.

Тесла також обладнана радаром з довгою хвилею опромінення, хвиля радару здатна проходити через несприятливі погодні умови, такі як: дощ, туман сніг та під другими машинами, грає значну роль в реагуванні на об’єкти які знаходяться спереду безпілотного авто. Ультра звукові датчики, які забезпечують круговий огляд всіх об’єктів навкруги навіть коли вони виникають раптово, значно розширюючи діапазон бачення авто, а також вони забезпечують систему корисною інформацією при паркуванні. Для обробки такої купи інформації в ТЕСЛА встановлено потужний центральний процесор обробки інформації, для зменшення потоку необроблених даних кожна камера оснащена своїм окремим мікропроцесором обробки даних, завдяки цьому тесла бачить одночасно у всіх напрямках і з не доступних людському оку ракурсах.

Головною фішкою безпілоту тесла являється технологія Tesla Vision. Вона використовує нейронні сітки, по з’явленню авторів вона більш надійно сприймає навколишнє середовище чим системи аналоги, які застосовують класичні методи обробки систем комп’ютерного зору. Конструктори тесла вважать, що за безпілотом з камерами майбутнє, адже потрібно дати можливість системі бачити так це бачить людина, з огріхом на те, що людина ніколи не зможе бачити на 360 градусів навкруги.

Рисунок 1.4. Візуально як бачить тесла

Інженери TESLA створили чудову та технологічну систему з використання великої кількості оптичних і ультразвукових датчиків, і зручною архітектурою системи. Чудово, що інформація частково опрацьовується на самих датчиках, це значно зменшує навантаження на головний комп'ютер системи. Важливо, що системи бачить навколишнє середовище так як його бачить людина, адже для створення систем які виконують велику кількість функцій на які здатна людина нам потрібно щоб система думала як людина, що реалізується методами ШІ, і бачила як людина. Звісно камери розпізнають образи використовуючи штучні нейроні мережі та алгоритми машинного навчання, підхід до вирішення задач за допомогою нейронних мереж хоч трудомісткий та дорогий, зате на виході ми отримуємо гнучку та адаптивну систему, яка може вирішувати широкий спектр задач.

Застосовувати камери можна і для управління технікою, камери мають широкий кут зору, вони коштують досить дешево і можуть застосовуватись у великій кількості. А з використанням систем на основі розпізнавання образів їх можна використовувати як датчики для переміщення у просторі, так і в якості сенсорів які сканують зовнішній вигляд об'єктів на будівництві чи на промисловому майданчику, з їх допомогою можна знаходити мітки на вантажах які будуть сигналізувати системі про гачки для підвісу або місця для заїзду з вилами для піднімання.

Lidar - лазерний далекомір

В компанії Google замість звичайних камер, використовують LiDar, лазерний далекомір, який встановлюється на кришу авто і генерує тривимірну карту простору навкруги в радіусі 100 метрів, отримавши дані, управляючий комп'ютер об'єднує їх з картами google, що дозволяє вчасно реагувати на небезпечні ситуації і притримуватись правил дорожнього руху.

Рисунок 1.5. Так виглядає Lidar на безпілотному авто від Google.

Як працює датчик, що має 360-градусний зір і точну інформацію про глибину? Простіше кажучи: датчик LIDAR постійно спрацьовує пучки лазерного світла, а потім вимірює, скільки часу потрібно, щоб світло повернулося до датчика. Випромінюючи мільйони пучків світла в секунду, вимірювання з датчика LIDAR дозволяють візуалізувати світ, який є справді 3D. Ви можете зробити точний вимір будь-якого об’єкта навколо [3]:

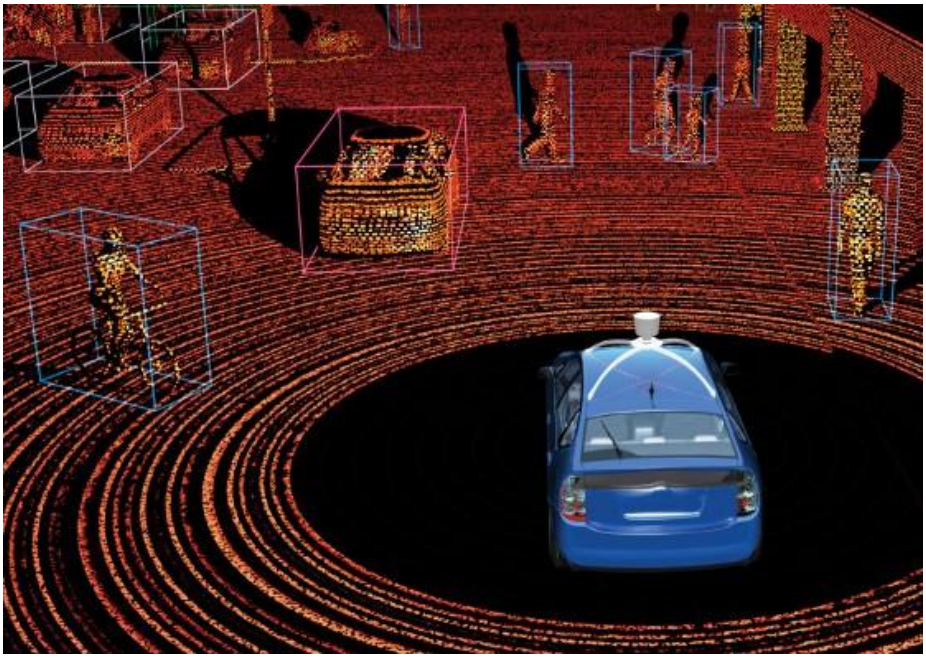

Рисунок 1.6. Так Lidar від Google бачить світ.

Радари, на безпілотному авто google їх чотири штуки, два на передньому, два на задньому бампері. Ця система використовує радіохвилі, для визначення дальності до об’єктів із використанням датчиків Delphi, і це перевірений метод, який дозволяє точно виявляти та відстежувати об’єкти на відстані 200 метрів, траєкторію та швидкість їх руху. Радар випромінюванням сканує територію і на основі відбитих імпульсів які потрапляють на антену формується карта загроз, яка стає очима авто і системою швидкого реагування на загрози, щось на кшталт бокових камер TESLA. Встановлено також передатчик GPS, який відслідковує положення автомобіля на маршруті його слідування та датчик положення, пристрій який допомагає визначити координати авто на карті. Відеокамера за дзеркалом заднього виду, відслідковує дорожні знаки та сигнали світлофору.

Алгоритм роботи безпілотного авто Google:

За дополотою лідар генерується об’ємна карта місцевості з кутом зору в 360 градусів та точною позицією усіх транспортних засобів в потоці, а керуючий комп’ютер з’єднує її з даними які містяться в пам’яті.

За допомогою GPS технологій розраховується також точне положення авто на місцевості, для уникнення пробок та складних дорожніх умов пов’язаних з неможливістю автопілоту самостійно рухатись в щільному потоці машин та на дорозі де кути поворотів надто великі.

На основі отриманих даних від радарів, камер та сенсорів спеціальний алгоритм оцінює ситуацію, генерує карту небезпек яка посекундно оновлюється для забезпечення повної безпеки руху.

Комп’ютер визначає траєкторію руху безпілотного авто, а також реагує на ситуацію на дорозі: рух других авто, жести поліцейського, їдучий спереду шкільний автобус, пішоходи, голо лід на трасі та масу других факторів. Для визначення факторів знаків та особливості місцевості використовуються камери та радари, по причині того, що лідар не може розрізнювати кольори.

Тож компанія Google використовує дещо другий підхід в створенні безпілотних авто, в своїх системах вони використовують лазерний далекомір – LIDAR. Системи на основі таких далекомірів формують карту місцевості навкруги автомобіля в тривимірному виді, це зручно адже така система точно визначає дистанцію до об’єктів і вона взагалі не підвладна погодним умовам. Вже сформована карта місцевості виступає гарним помічником коли автомобіль повертається назад, адже він може послуговуватись нею і направити всі обчислювальну потужність на прогнозування небезпечних ситуацій. Натомість технологія не в повній мірі може виконувати всі функції безпілотного авто за рахунок одного тільки LIDAR, вона не може відрізняй кольори, тому не може прочитати дорожні знаки або сигнали світлофору, звісно застосування розпізнавання образів теж неможливе в повній мірі.

Лідар використовуються не тільки в якості встановлення на автомобілі, цим пристроєм можна оснастити вертоліт або літак, та створити карту майданчику по якому буде перемішуватись безпілотна техніка в промисловості. З’єднавши її з мапою території та даними від датчиків GPS, можна отримати високоточну мапу місцевості.

1.5 Технології безпілотного керування в промисловості

В американській компанії Kawasaki Construction Machinery of America Corp. Створили фронтальний навантажувач лінійки Z7 модель 70Z7, усі навантажувачі цієї серії оснащені системою управління з елементами штучного інтелекту IntelliTech, яка має можливість розумного набору матеріалу в ківшу, підсистема IntelliDig за рахунок оптимізації відношення величини колісної тяги і зусилля копання, забезпечує роботу гідро механізму стріли та ківшу. Система також передбачає пришвидшене виконання операцій QuickCycle і автоматичне переключення передач FlexShift в залежності від умов руху. Те так званий 1 рівень автоматизації, відсутність автопілоту з використанням систем облегшення роботи оператора.

Роботизовані навантажувачі використовуються при роботі на складах. Майбутнє за роботизованою складською технікою, вона дає найбільшу економію грошей при опрацюванні більшої кількості вантажів при виконанні операцій, що повторюються. Це лінійки техніки для логістики всередині підприємства. Робота складської техніки без операторів – як однієї машини так і всього парку загалом, можлива у всіх галузях промисловості, торгівлі.

Рисунок 1.7. Роботизований навантажувач Linde L-MATIC AC

Інтеграція навантажувача з системою управління парком техніки конкретного складу забезпечує система робо-менеджер, цей програмний застосунок в режимі реального часу збирає, опрацьовує та передає данні про роботу навантажувача і створює взаємодію з другими навантажувачами даного парку, а також обладнанням та механізмами. Ця інтелектуальна система також слідкує за передаванням даних необхідних для проведення своєчасного технічного обслуговування.

В порівнянні з класичними системами автоматизації навантажувачів, які для орієнтування по складу використовують системи на основі лазерних рефлекторів, навантажувачі Linde не потребують спеціальної інфраструктури. Це стало можливим з використанням технологій гео-навігації. При використанні штабелера L-MATIC, якщо робо-менеджер дає задачу змінити стелажну алею, то він рухається у відношенні з даними о конструкційних елементах будівлі, стінах, колонах, стелажах. Заздалегідь внесені в нього данні о конструкційних елементах гарантують безпечність і надійність пересування. Потрапивши в потрібну алею, штабер рухається індуктивним або механічних направляючим, які забезпечують безпечну роботу на великих швидкостях.

Застосування безпілотної техніки на виробництві це безперервна та цілодобова робота без перерви на обід, сон та відпустку. Безпілотна техніка в промисловості це не тільки можливість зекономити на фонді оплати труда операторів та інших робітників, але і багатократне збільшення продуктивності роботи промисловості, точності і якості опрацювання вантажів в великих об’ємах. Вже були створені системи які можуть працювати на території закритого складу, за основу взяті звичайні навантажувачі, вони оснащені системами на основі камер і вільно перемішуються по території складу перевозлячи коробки та полети. Вони безпечні бо можуть швидко відреагувати на умови, що різко змінюються, наприклад вони миттєво зупиняються перед предметом який упав на маршрут по якому вони слідували і коли перепону забрати вони спокійно можуть продовжити свій рух. Безпілотна техніка здатна працювати без операторів і не прив’язана до щоденного графіку роботи підприємства, високоточне маневрування на відміну від керування людиною в рази знижує ризики створення аварійних або травмонебезпечних ситуацій. Тільки в одних США щорічно проходять дев’яносто шість тисяч подій з вилочними навантажувачами, десятки з таких випадків призводять до трагічних наслідків і обходяться компаніям приблизно в тридцять мільярдів доларів збитків. Тобто реалізація високоточних автономних систем які до того ж фіксували б в базі всі операції та події могло б зекономити компаніями велику кількість грошей і зменшити фон промислового травматизму.

Висновки до першого розділу:

У цьому розділі ми розглянули принципи та алгоритми побудови безпілотних систем, визначили якісні характеристики якими повинні володіти системи. Розглянули технології якими послуговуються розробники таких систем. Головними складовими системи автопілоту є система орієнтування в просторі, алгоритм прийняття рішень та карта якою користується система по мірі руху.

Тотальна глобалізація, ріст виробництва та споживання змушує промисловість пристосовуватись, в таких умовах системи на основі штучного інтелекту виступають гарним замінником праці людей, так як вони можуть виконувати однотипну роботу з прогнозованою ефективністю, не бояться переробітку, а інвестиції в них швидко відбиваються.

Системи які використовуються в перевезенні вантажів основуються на принципах вбудованої інфраструктури в закритих приміщеннях, що є малоефективним для вирішення проблем на великих площах промислових складів. Також системи з використанням викових навантажувачів існують в просторі складів з чітко окресленими рамками і використовують маркування позицій вантажів з бази даних, це є гарним рішення для створення системи в закритому просторі приміщення складу, але для великої території складу порту воно не підходить. Адже там типи вантажів зовсім другі, і використання таких систем малоефективне.

Стоїть необхідність створення систем які б могли функціонувати використовуючи більш важку і складну техніку, перемішуватись по території закритого комплексу і виконувати роботу яку може виконувати персонал ефективніше та при цьому мати високу безпечність.

Задача розробити систему безпілотного керування і автоматизації роботи техніки на території промислового майданчику, з використанням вже відомих технологій по автоматизації та безпілотному керуванню.

2 ОГЛЯД СИСТЕМИ РОЗПІЗНАВАННЯ ОБРАЗІВ

2.1 Постановка задачі

Для створення системи в якій може функціонувати безпілотний апарат, потрібно вирішити кілька ключових задач. Створити систему логістики вантажів по території порту і перенести її в зрозуміле для агенту середовище, таки системами зазвичай виступають бази даних під керуванням системи управління базами даних, розташувати їх можна як локально на території підприємства, так і на віддаленому сервері у хмарі, тоді потрібно використовувати широко полосне підключення до інтернету і орендувати виділений сервер на якому вже і будуть розташовані база даних, СУБД, програмне забезпечення. Потім потрібно створити інтерфейс який зможе опрацьовувати вантажі, як приклад такого інтерфейсу може виступати система на основі візуальних міток які може зчитувати як машина, так і людина з допомогою спеціальних сенсорів. Дані які будуть отримані таких шляхом будуть передаватись в систему управління, вона в свою чергу буде проводити маніпуляції з базами даних, визначати точку в яку потрібно перемістити вантаж, які маніпуляції з ним потрібно провести або просто заносити його в базу для ведення документації. Також потрібно створити карту в якій може орієнтуватись безпілотний навантажувач, вона повинна включати в себе сітку доріг по яким буде перемішуватись безпілотний апарат та зони в яких він буде виконувати роботу. Це потрібно для більш просторого організування роботи системи, умовно поділивши територію на зони буде простіше здійснювати логістику вантажів по території, в зони де знаходяться вантажі можна поділити на сектори і дати їм номери або умовні кодові імена, занести цю інформацію в базу і використовувати для орієнтування. Дороги по яким рухаються безпілот можна позначити лініями для того щоб він точно візуально міг визначити рамки в яких він може рухатись, а також щоб інша техніка та люди не заважала переміщенню безпілоту. Також потрібно створити систему за допомогою якої навантажувач

може маніпулювати вантажами в контейнерах з мінімальною участю людини в процесі, можна нанести на вантажі з однаковими габаритами такими як контейнери помітки які буде розпізнавати системи і використовувати програму поведінки яку можна прописати раніше, а з негабаритними вантажами можна буде використовувати віддалений пульт керування з оператором якій підчепить вантаж і далі система продовжить виконання в штатному режимі. Сам безпілот буде рухатись по території під управлінням команд які буде надавати штучний інтелект системи на основі прописаної моделі поведінки реагуючі на події які відбуваються навколо за допомогою сенсорних камер, які розпізнають образи за допомогою комп'ютерного бачення, вимірюють дистанцію зближення до об'єктів за допомогою радарів.

2.2 Автопілот

Задача автопілоту на промисловому майданчику максимально продуктивно та безпечно перемішуватись з місця на місце, та виконувати по ходу переміщення корисні дії, перевозити та пересувати вантажі, букерувати причепи, підвішувати та навантажувати вантажі на другу техніку. Для цього звісно потрібно навчити автопілот бачити навколишній світ, оскільки тільки у випадку коли автопілот розуміє що попереду нього і куди йому рухатись, він зможе почати діяти та виконувати поставлені перед ним задачі.

Частково вирішити проблему орієнтування в просторі можна створивши ефективну карту місцевості, яку зможе прочитати автопілот, так я постійна робота датчиків генерує величезну кількість інформації яку потрібно опрацьовувати, а значить система буде потребувати величезної обчислювальної потужності. . На карті можна позначити всі маршрути по яким буде рухатись техніка під управлінням ШІ, створити своєрідну сітку маршрутів з прилягаючими до них зонами, такими як: зона відвантаження, зона завантаження, зона стоянки в спокійній позиції або ангар для нічного паркування, зона технічного обслуговування. Так як на промислових майданчиках ці території часто розділені

між собою великою відстанню і знаходяться в різних іноді перпендикулярних точках простору, створення такої мапи потягне за собою створення алгоритму найкоротшого шляху, для ефективного використання системою робочого часу та паливо-мастильних ресурсів

Система глобального інформування, в такій системі усі юніти які знаходяться в глобальній мережі повідомляють один одного про своє положення, про ситуації які виникають на маршруті слідування агентів під управлінням ШІ, наприклад може скластися ситуація в якій умовна поламана вантажівка перегороджує дорогу яка занесена в карту переміщення агенту, тоді агент повідомляє усім юнітам в системі, що дорога перекрита і шукає обхідний шлях, в такому випадку не буде складатись ситуація в якій кожний агент буде впиратись в перекритий шлях і марно витрачати час роботи та ресурси компанії, сама ж точка з перекритим шляхом буде позначатись на табло операторів системи, які можуть по камерам або візуально визначити який транспорт, або який об’єкт заважає нормальній роботі системи, усунувши проблему та оцінити чи вільна дорога тепер, вони можуть зняти помітку заблокованого маршруту з мапи, система при цьому увесь час буде працювати в нормальному режимі. Такі прораховані шляхи вирішення проблем підвищують відмово стійкість системи та роблять її більш пріоритетною в очах замовників.

2.3 Розробка AS-IS моделі управління транспортним засобом

Модель системи у вигляді діаграми AS-IS, представляє собою наглядну реалізацію роботи з різними алгоритмами системи:

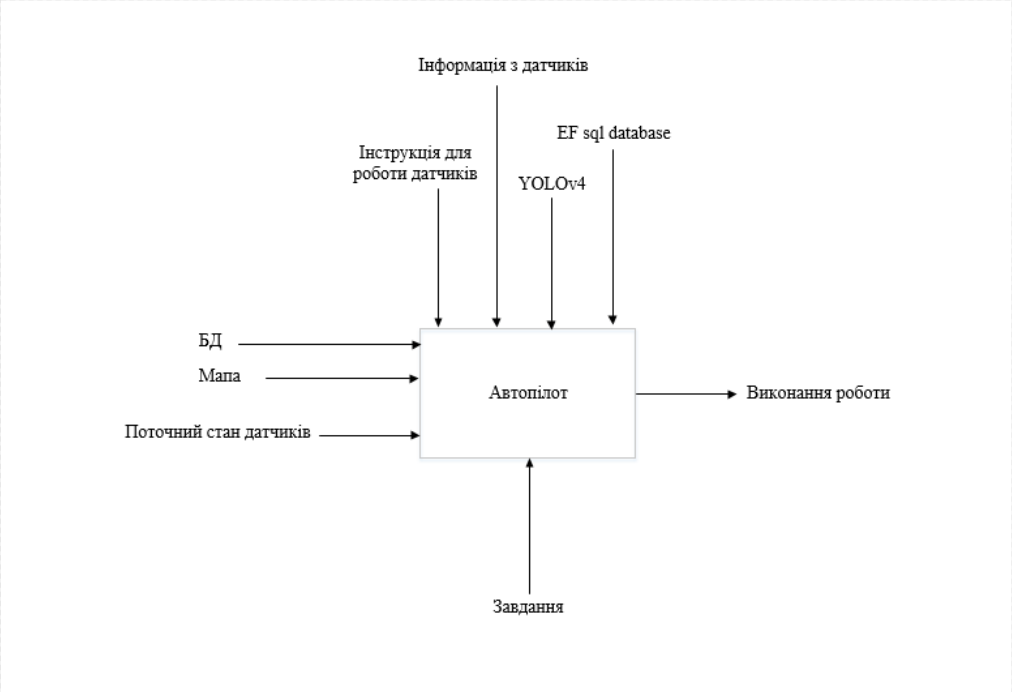

Рисунок 2.1 Модель AS-IS

Модель працює наступним чином: коли система запускаються вона від’єднується до датчиків агенту, при цьому підключаються до бази даних з якої бере список завдань та під'єднується до мапи з якої бере інформацію про положення вантажів. Система береться за виконання першого доступного в списку завдання переходячи до наступного тільки після виконання, при виконані автопілот послуговується інструкціями по роботі з датчиками та нейронною мережею YOLOv4 для розпізнавання образів, всі дії агенту документуються та записуються в базу даних.

2.4 Система розпізнавання образів

Система розпізнавання образів потрібна як для точного, швидкого та безпечного переміщення в просторі, так для розпізнавання вантажів і поміток на них, таких як наприклад QR коди.

Існує велика кількість методів розпізнавання об’єктів на зображенні, вибір конкретного методу зумовлений, власними та особливими характеристиками об’єкту який потрібно розпізнати. Коли задача з розпізнавання ставиться

неформальним чином або властивості об’єкта який ми шукаємо задаються без жорстких математичних рамок. Для розрішення таких задач, потрібно створити сформулювати властивості потрібного об’єкта і створити стійкий метод розпізнавання об’єкту, що шукаємо. Для вирішення задачі потрібно знайти, зібрати, узагальнити, та сформулювати в математичних термінах емпіричні спостереження. Головна проблема проявляється в тому, що не можливо описати усі якості предмету або не всі якості можуть відповідати всім об’єктам. Тому в процесі нормалізування допускаються спрощення, які в результаті, знижують якість алгоритму. Тому важливо зберегти баланс важкості розрахунків та бажаної точності.

Система розпізнавання образів – електронно-обчислювальний комплекс, здатний моделювати розумові процеси, властиві людині під час прийняття рішень, з метою виявлення аналогій серед досліджуваних об’єктів. Для розпізнавання образів необхідно розв’язати дві основні задачі: розбиття простору ознак розпізнавання на області, що відповідають певному класу об’єктів, та визначення належності реалізації образу, що розпізнається, до відповідного класу. Найчастіше система розпізнавання образів за аналогією з біонічною системою функціонує в двох режимах: навчання й екзамену. На етапі навчання з метою побудови вирішальних правил здійснюється розбиття простору ознак на класи розпізнавання [5].

Головними критеріями якості ознак для вирішення широкого спектру задач зв’язаних з розпізнаванням образів, в тому числі зорових, являється розділення параметрів ознак, а також складність їх отримання. Також враховується важливість того, що потрібно якомога швидше знайти область потрібного об’єкту. Ця можливо тільки при класифікації більшого числа елементів при обробці кожного зображення.

Для вирішення задач, зв’язаних з розпізнаванням зручно використовувати досить прості алгоритми отримання ознак, на приклад використання алгоритмів розпізнавання на основі примітивів Хаара. Примітиви Хаара представляють собою результати порівняння якості в двох прямокутних областях [14].

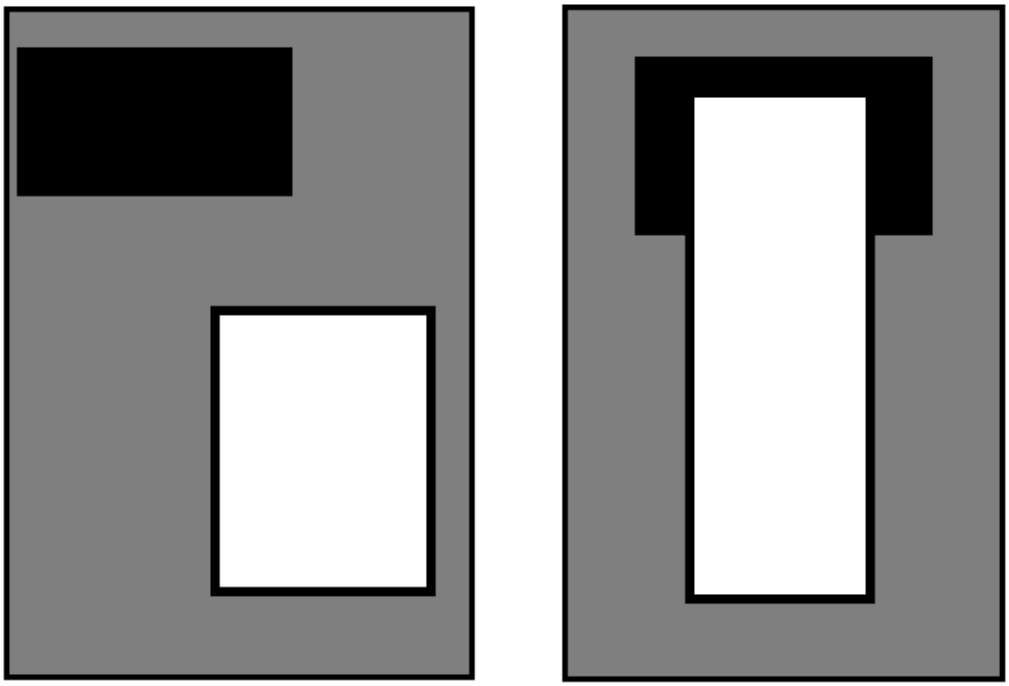

Рисунок 2.2. Примітиви Хаара

Ознаки які використовуються для розпізнавання об’єктів: А) – області пересікаються, Б) – області не пересікаються.

Ознаку для даної області на отримання якостей можна розщитати використовуючи формулу: (2.1).

$$R = \frac{S_б}{N_б} - \frac{S_ч}{N_ч}, \qquad (2.1)$$

Для областей, що не пересікаються на рисунку (2.1) варіант А, використовується формула (2.2):

$$R = \frac{S_б}{N_б} - \frac{S_ч - S_{ч \cap б}}{N_ч - N_{ч \cap б}}, \qquad (2.2)$$

Де Ч і Б – це Чорна і Біла області відповідно;

S – це сума яскравості пікселів які знаходяться під областю зображення;

N – це кількість пікселів які знаходяться в цій області;

ч ∩ б – пересічення областей.

Таким способом отримуємо відклики, які означають різницю середніх яскравості пікселів, які знаходяться в зображенні яке розпізнається, в білій та чорній області.

Перевагою відкликів схожих якостей в тому, що вони не залежать від масштабування зображення і від змішування кольорових схем і яскравості зображення. Крім вище вказаних формул, для обрахування відкликів на вказану область зображення можуть використовуватись наступні формули:

1) У випадку областей, що не перетинаються:

$$R = S\text{б} - N\text{ч}, \tag{2.3}$$

2) Для областей, що пересікаються:

$$R = S\text{б} - (S\text{ч} - S\text{ч} \cap \text{б}), \tag{2.4}$$

3) У випадку областей, що пересікаються:

$$R = \begin{cases} 1, \frac{S\text{б}}{N\text{б}} > \frac{S\text{ч}}{N\text{б}} \\ -1, \frac{S\text{б}}{N\text{б}} \leq \frac{S\text{ч}}{N\text{ч}} \end{cases}, \tag{2.5}$$

4) У випадку областей, що не пересікаються:

$$R = \begin{cases} 1, \frac{S\text{б}}{N\text{б}} > \frac{S\text{ч} - S\text{ч}\cap\text{б}}{N\text{ч} - N\text{ч}\cap\text{б}} \\ 1, \frac{S\text{б}}{N\text{б}} \leq \frac{S\text{ч} - S\text{ч}\cap\text{б}}{N\text{ч} - N\text{ч}\cap\text{б}} \end{cases}, \tag{2.6}$$

Якщо дивитись з позиції інваріантності можливих значень властивостей перетворення яскравості зображень, використання формул (2.5, 2.6), для визначення значень відкликів являється найбільш доцільним.

Використовуючи ці вирази, отримуємо значення властивості, які будуть симетричними до будь-яким можливим лінійним перетворенням (монотонно зростаючим) яскравості, за умови, що перетворення не змінять класову приналежність, але допускається значна зміна яскравості розподілу. До подібних перетворень відносять зміна контрасту і яскравості, що застосовуються при обробці фото- та відео в багатьох пристроях, для підвищення якості зображення.

Існує такий підхід до вирішення завдань розпізнавання (класифікації) як посилення простих класифікаторів. Підхід заснований на поєднанні декількох простих класифікаторів в один сильніший. Ефективність класифікатора - сила, це якість вирішення поставленого завдання класифікації.

В основі методів розпізнавання, заснованих на посиленні простих класифікаторів, знаходиться наступна ідея [6]: поєднати кілька простих (елементарних) ознак для того, щоб скомбінований ознака вийшов більш потужним. Розглянемо приклад з автомобільними гонками: нехай людина, знайомий зі світом автоспорту вирішить розробити програму, що допомагає визначати переможця і пророкує шанси на перемогу різних спортсменів. В результаті перегляду деякої кількості змагань і провівши опитування глядачів, які роблять ставки, ця людина виділив кілька Імперична правил:

- Потрібно ставити на машину, яка перемогла в чотирьох попередніх колах;
- Потрібно ставити на машину, на яку максимальну кількість ставок;
- Фаворитами завжди є чемпіони попередніх гонок і т. Д.

Зрозуміло, що будь-який з перерахованих вище правил ненадійно і використовувати ці правила окремо недоцільно.

Тому, виникає необхідність оптимально скомпонувати ці правила для досягнення надійного результату. Набір алгоритмів, що працюють з використанням методу посилення простих класифікаторів, дозволяє зібрати

прості ознаки таким чином, що б отримати один, але сильніший ознака. Розглянемо один з перших алгоритмів з цієї групи - AdaBoost.

На основі цього алгоритму була побудована на даний момент найбільш ефективна за якістю розпізнавання і швидкості виконання послабленной завдання система розпізнавання об'єктів на зображенні - Viola-Jones ObjectDetector [10]. У числі основних достоїнств методів AdaBoost можна відзначити:

- Висока швидкість роботи;
- Висока ефективність (точність);
- Простота реалізації.

Після присвоєння значення ваги кожного елементу навчальної вибірки задається рівень важливості саме цього прикладу для кожного кроку навчального алгоритму. Залежно від ваги елемента, алгоритм прикладає різні «зусилля», чим більше вага, тим сильніше алгоритм намагається класифікувати цей приклад як правильний.

Рисунок. 2.3. Контейнер з розпізнаними образами.

В основі алгоритму Хафа лежить метод виявлення ліній на зображенні. Використання цього методу дає можливість задавати параметри пошуку ліній або сімейства кривих і знаходити на зображенні безліч кривих даного сімейства.

2.4 Нейрона мережа по розпізнаванню образів YOLO

Оскільки YOLO дивиться на зображення тільки один раз, то плаваюче вікно – це неправильний підхід. Замість цього, усі зображення розбиваються за допомогою сітки на зображення розміром $S * S$. Після того як кожна комірка відповідає за передбачення декількох речей

По перше, кожна комірка відповідає за передбачення декількох змістових рамок і показника впевненості (confidence) для кожного з них – другими словами, це можливість того, що дана рамка містить об’єкт. Якщо в якійсь комірці сітки об’єктів немає, то дуже важливо, confidence щоб для даної комірки був дуже малим.

Коли ми візуалізуємо усі ці передбачення, ми отримаємо мапу усіх об’єктів та набір частин зображення, що містяться в рамках, які ранжовані по їх confidence:

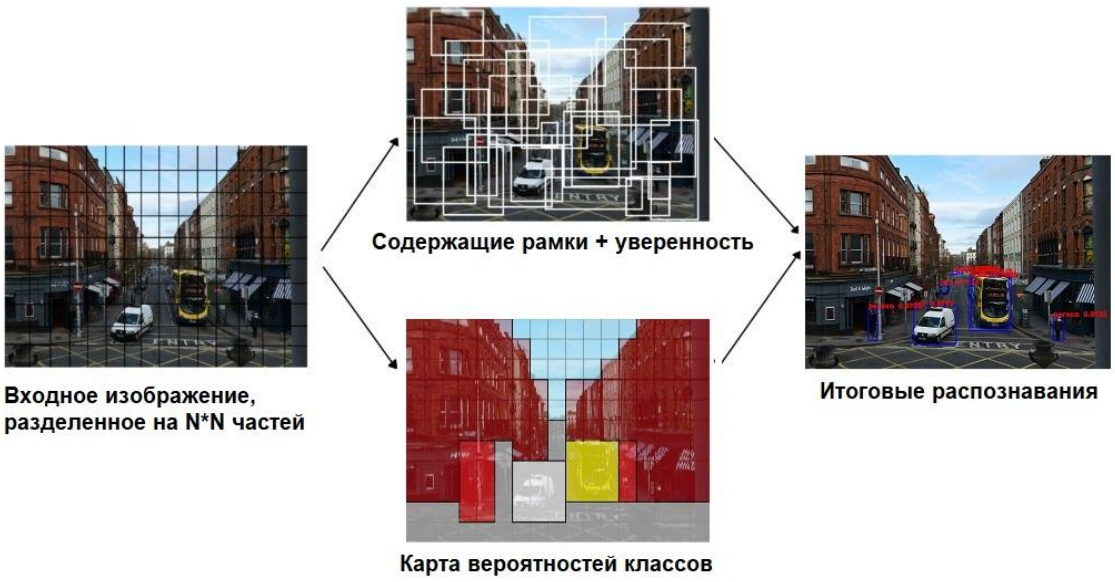

Рисунок. 2.4. Мапа з ранжуванням об’єктів по confidence.

По-друге, кожна клітинка відповідає за передбачення ймовірностей класів. Це не означає, що якась комірка містить якийсь об'єкт, це всього лише ймовірність. Таким чином, комірка мережі пророкує автомобіль, але це означає, що якщо там є якийсь об'єкт, то це автомобіль.

У YOLO для передбачення рамок, що містять об’єкти, використовуються якірні рамки (anchor boxes). Їх основна ідея полягає в приречення двох різних рамок, які називаються якірними рамками або формою якірних рамок. Це дозволяє нам асоціювати два передбачення з цими якірними рамками. Загалом, ми можемо використовувати і більшу кількість якірних рамок (п'ять або навіть більше). Якоря були розраховані на датасета COCO за допомогою k-means кластеризації [11].

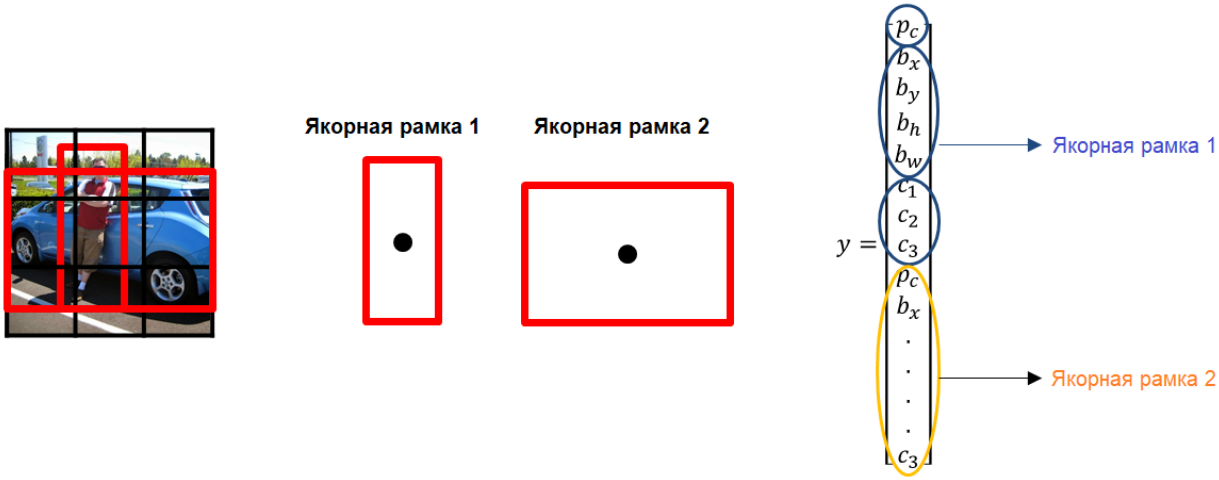

Рисунок 2.5. ділення зображення на якірну сітку

У нас є сітка, кожна комірка якої повинна передбачувати:

- для кожної містить рамки: 4 координати (t_x, t_y, t_w, t_h) та одну «помилку об’єктності» тобто нашу координату confidence, яка визначає є об’єкт чи ні.
- деяку кількість ймовірностей класів.

Якщо є деякий зсув щодо верхнього лівого кута C_x, C_y, то ці передбачення відповідають наступним формулам:

$$
\begin{aligned}
b_x &= \sigma(t_x) + C_x \\
b_y &= \sigma(t_y) + C_y \\
b_w &= \rho_w e^{t_w} \\
b_h &= \rho_h e^{t_h}
\end{aligned}
\tag{2.7}
$$

де p_w та p_h відповідають ширині та висоті рамки в якій знаходиться потрібний об’єкт. Замість передбачення зміщень, як це було в другій версії YOLO, автори передбачують координати локації відносно положення комірки сітки.

Це вивід нашої нейронної мережі, усього там $S * S[B * (4 + 1 + C)]$ виводів, де B – це кількість рамок, передбачення кожної комірки залежить від того, в скількох масштабах рамок ми хочемо робити наші передбачення, C – кількість класів, 4 – кількість рамок, а 1 – передбачення об’єкту. За один прохід ми можемо пройти від вхідного зображення до вихідного тензору, який відповідає розпізнаним об’єктам зображення, також можна пригадати, що YOLO v3 передбачує рамки у трьох різних масштабах.

Тепер, якщо ми візьмемо ймовірності і помножимо їх на значення confidence, ми отримаємо всі рамки, зважені по їх ймовірності утримання цього об'єкту.

Просте порівняння з порогом дозволить нам позбутися від пророкувань з низькою confidence. Для наступного кроку важливо визначити, що таке перетин щодо об'єднання (intersection over union). Це відношення площі перетину прямокутників до площі їх об'єднання:

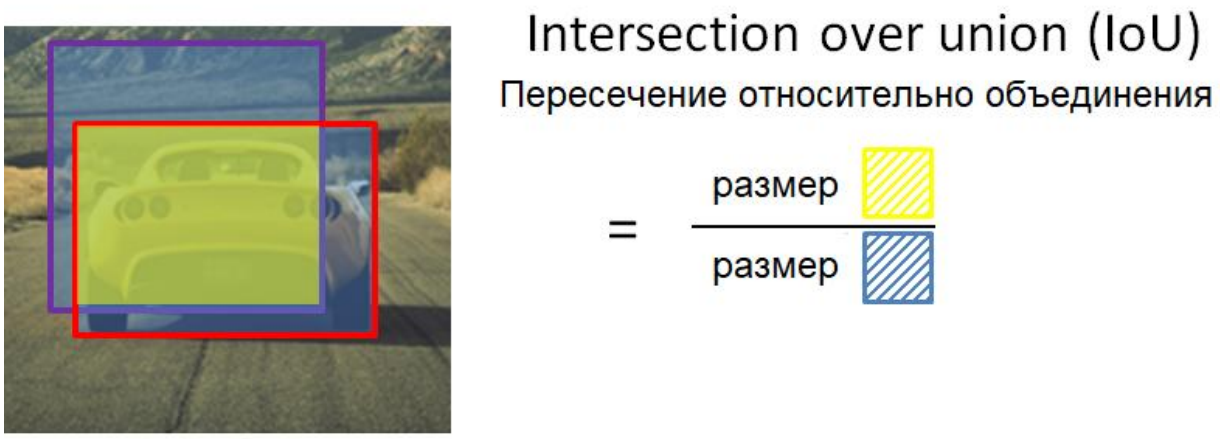

Рисунок 2.6 перетин щодо об’єднання

Після цього у нас ще можуть бути дублікати, і щоб від них позбутися, ми застосовуємо придушення НЕ-максимумів. Придушення НЕ-максимумів бере вміст рамки з максимальною вірогідністю і дивиться на інші вмісти рамки, розташовані близько до першої. Найближчі рамки з максимальним перетином щодо об'єднання з першої рамкою будуть придушені.
Оскільки все робиться за один прохід, модель працює майже з такою ж швидкістю, як класифікація. Крім того, всі прогнози виробляються одночасно, а це значить, що модель неявно вбудовує в себе глобальний контекст. Простіше кажучи, модель може засвоїти, які об'єкти зазвичай зустрічаються разом, відносні розміри і розташування об'єктів і так далі.

Отже, існує кілька архітектур нейронних мереж, створених для визначення об'єктів. Вони в основному поділяються на «дворівневі», такі як RCNN, fast RCNN і faster RCNN, і «одноровневі», такі як YOLO.
«Дворівневі» нейронні мережі, перераховані вище, використовують так звані регіони на зображенні, щоб визначити, чи знаходиться в цьому регіоні певний об'єкт.
Зазвичай це виглядає так (для faster RCNN, яка є найшвидшою з перерахованих дворівневих систем):

Подається картинка / кадр на вхід

Кадр проходить через CNN для формування feature maps

Окремою нейронною мережею визначаються регіони з високою ймовірністю знаходження в них об'єктів

Далі ці регіони за допомогою RoI pooling стискаються і подаються в нейронну мережу, що визначає клас об'єкта в регіонах

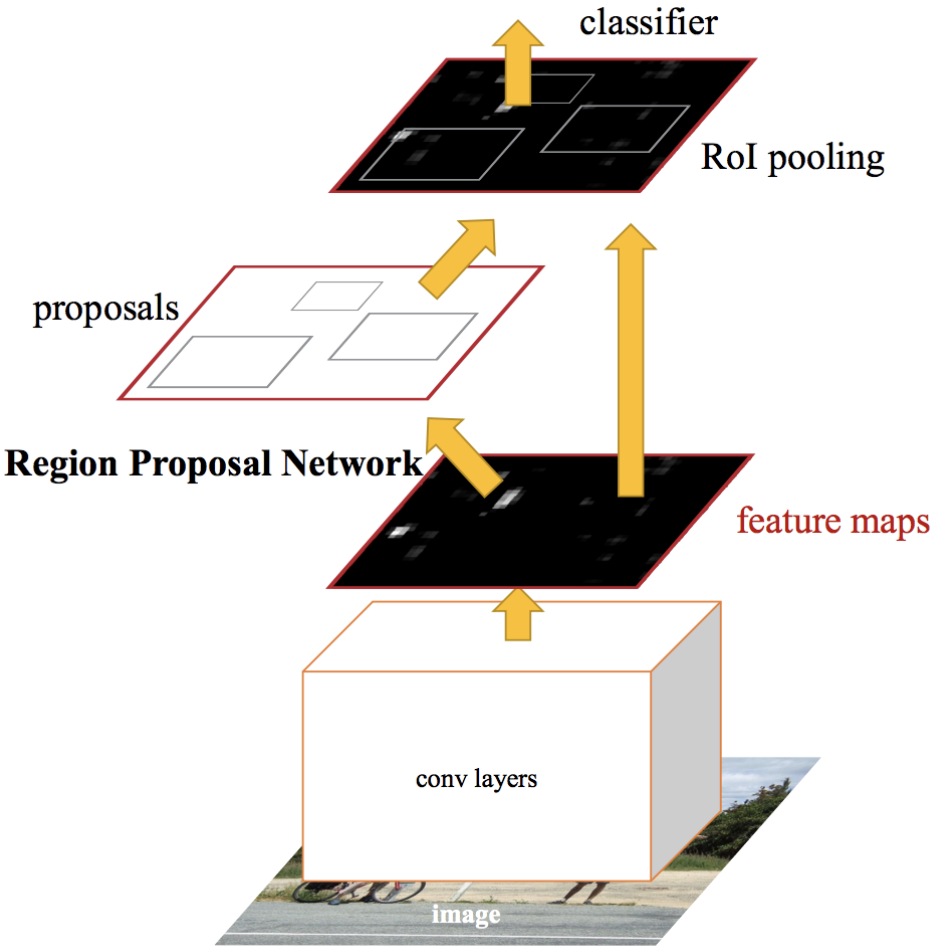

Рисунок 2.7 Прогонка зображення через чотири етапи для класифікації

Але в цих нейронних мережах є дві ключові проблеми: вони не дивляться на картинку «повністю», а тільки на окремі регіони, і вони відносно повільні.

У чому ж крутість YOLO? У тому, що ця архітектура не має двох проблем понад, і вона довела неодноразово свою ефективність.

Взагалі архітектура YOLO в перших блоках не сильно відрізняється по «логікою блоків» від інших детекторів, тобто на вхід подається картинка, далі створюються feature maps за допомогою CNN (правда в YOLO використовується своя CNN під назвою Darknet-53), потім ці feature maps певним чином аналізуються, видаючи на виході позиції і розміри bounding boxes і класи, яким вони належать [12].

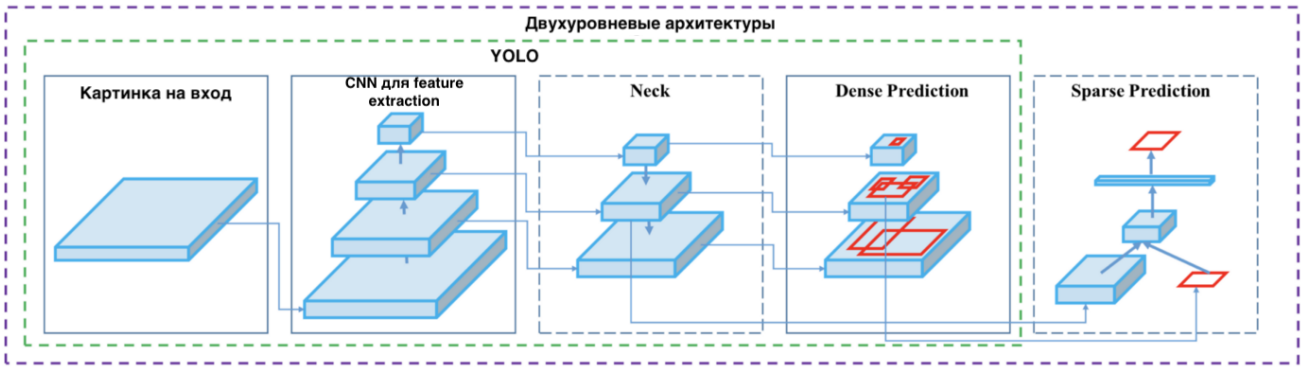

Рисунок 2.9 Дворівнева архітектура

Але що таке Neck, Dense Prediction і Sparse Prediction?

З Sparse Prediction ми розібралися трохи раніше - це просто повторення того, як дворівневі алгоритми працюють: визначають окремо регіони і потім класифікують ці регіони.

Neck (або «шия») - це окремий блок, який створений для того, щоб агрегувати інформацію від окремих шарів з попередніх блоків (як показано на малюнку вище) для збільшення акуратності передбачення. Якщо Вас зацікавила ця - можете погуглити терміни «Path Aggregation Network», «Spatial Attention Module» і «Spatial Pyramid Pooling». І, нарешті, те, що відрізняє YOLO від всіх інших архітектур - блок під назвою (на нашій картинці вище) Dense Prediction. На

ньому ми сфокусуємось трохи сильніше, тому що це дуже цікаве рішення, яке якраз дозволило YOLO вирватися в лідери по ефективності визначення об'єктів. YOLO (You Only Look Once) несе в собі філософію дивитися на картинку один раз, і за цей один перегляд (тобто один прогін картинки через одну нейронну мережу) робити всі необхідні визначення об'єктів. Як це відбувається? Отже, на виході від роботи YOLO ми зазвичай хочемо ось це:

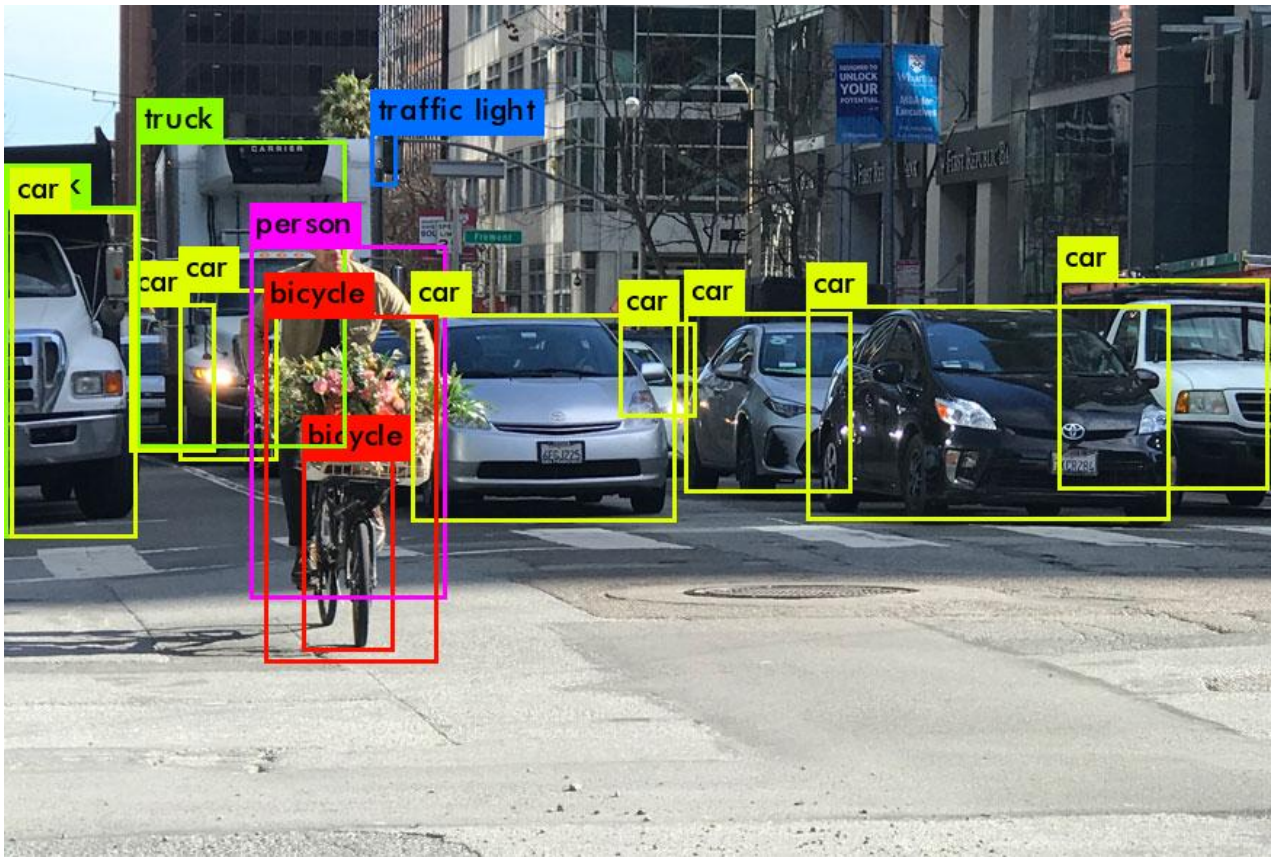

Рисунок 2.10 Зображення після розпізнавання за допомогою YOLO

Бачимо зображення коли нейронна мережа YOLO вчилась на даних, загалом технологія досить оптимізована та швидко працююча, для її використання у проекті потрібно навчити мережу розпізнавати специфічні образи, такі як: навантажувачі, контейнери з помітками, крани, місця відвантаження. Для цього потрібно просто навчити мережу на сетах з картинками яких спочатку потрібно набрати і підготувати, потім потрібно буде просто додати пул вагів для розпізнавання таких образів до бази.

2.5 Розробка схем роботи бази даних та СУБД

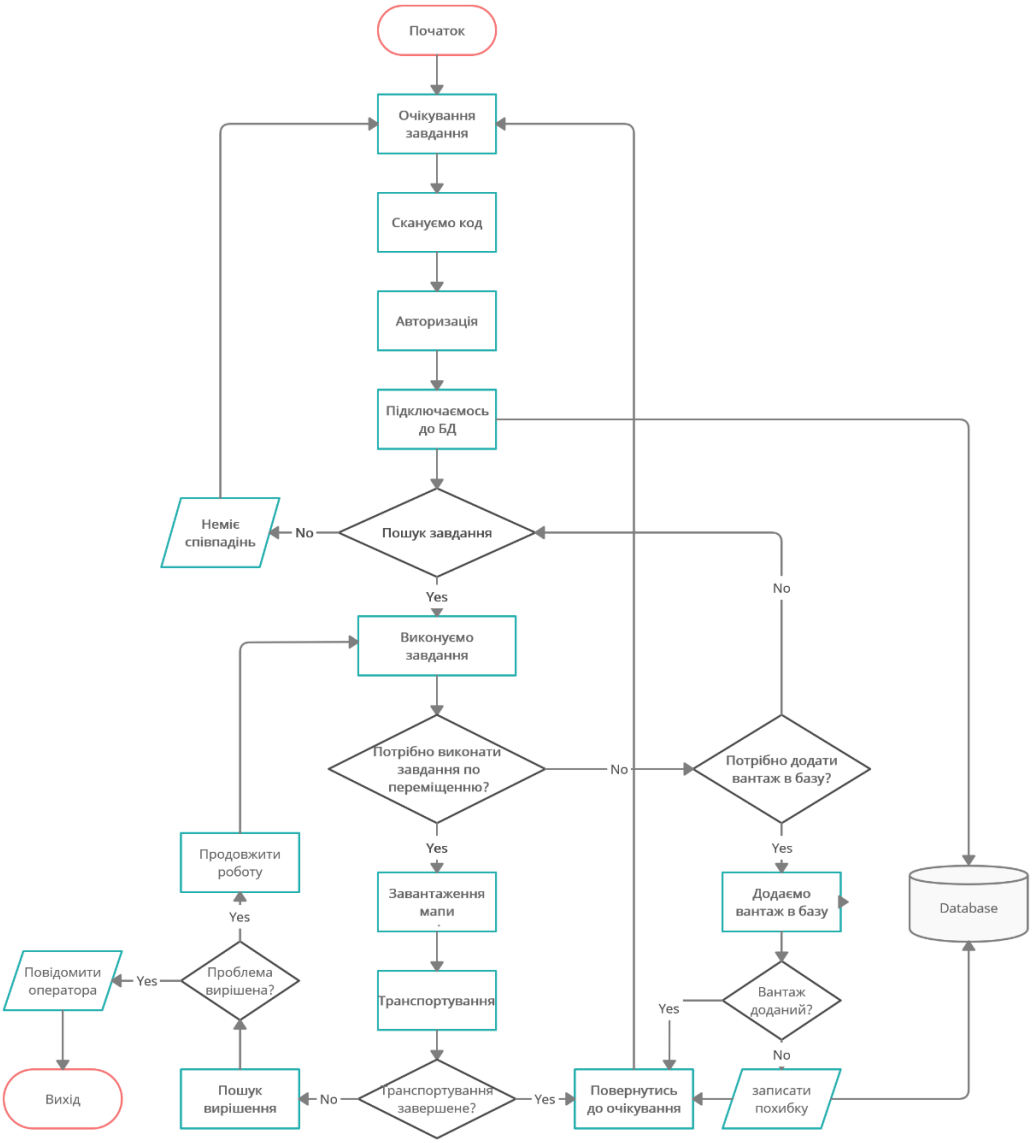

Рисунок 2.11. Блок-схема алгоритм роботи бази даних

База даних в якій буде зберігатись інформація про всі вантажі на території повинна містити в собі таблиці з даними про територію складу. Які місця заповнені вантажами, які пустують, кожний запис повинен містити станичну інформацію про вантаж, його вміст, власника, дату запакування та динамічну інформацію: дату його прибуття на склад, місце його перебування. Ця інформація повинна вноситись в базу на основі правил, якщо проводяться маніпуляції з

вантажем вони повинні бути збережені в базу даних. Це дозволяє біль легко вести бухгалтерію і при цьому можна збирати статистику і покращувати роботи системи для збільшення продуктивності роботи ШІ, та програмних алгоритмів, роботи порту загалом. БД

СУБД такої бази даних буде отримувати інформацію від персоналу або від самих агентів про поступлення нового вантажу за допомогою сканування QR коду яркий буде розміщено на боковій стороні контейнеру, в такий код буде записаний індикаційний номер вантажу

База буде включати в себе такі таблиці як:

Operator – оператор який сидить за комп’ютером і слідкує за похибками в системі, їх в штаті може бути декілька, всі вони виконують однакову роботу, слідкують за моніторами з готовністю знаходити вирішення нестандартній ситуації з якої не може вийти ШІ.
Agent – це таблиця з усіма видами техніки під управлінням штучного інтелекту на території, в базі їх буде від одного до декількох сотень, кожний такий агент буде мати свій конкретний тип та свою спеціалізацію, буде заточений під виконання свої конкретних завдань.
Work – таблиця в яку заносяться всі роботи які потрібно виконати, вона матиме прив’язку до території, прив’язку до певної техніки яка може виконувати маніпуляції з вантажем такого типу, прив’язку по часу за який потрібно виконати операції, пріоритет операцій.
Map – таблиця буде зберігати в собі шляхи до створених раніше моделей мап, записувати в себе місцеположення техніки яка перемішується по комплексу зберігати в собі сітку доріг по яким буде рухатись ШІ, в неї також буде записана карта зон в яких можуть проводитись різні операції: завантаження, відвантаження, технічне обслуговування тощо. Ці зони завантаження та

відвантаження будуть поділені на сектори, в базі також будуть записані дані про те які саме вантажі зберігаються в секторі.

Operation – це список операцій які потрібно провести, це стек який заповнюється знизу та виконується зверху, сюди потрапляють усі операції завантаження, відвантаження, переміщення.

Cargo – таблиця з вантажами, в цю таблицю потрапляють всі, щойно від скановані вантажі, також сюди потрапляє інформація про власника і того хто зберігає вантаж, коли він був запакований і чим.

Errors – таблиця з системними похибками, відслідковування помилок являється важливою частиною вдосконалення та нормального функціонування будь якої системи. Сюди потраплятимуть дані про помилки від техніки яка знаходиться під управлінням ШІ, оператор буде відслідковувати помилки і вчасно втручатись в роботу за необхідності.

Container – це дані про контейнери, можна сказати що це всього лиш інформація про серійні номери контейнерів які використовувались при запакуванні в них вантажів.

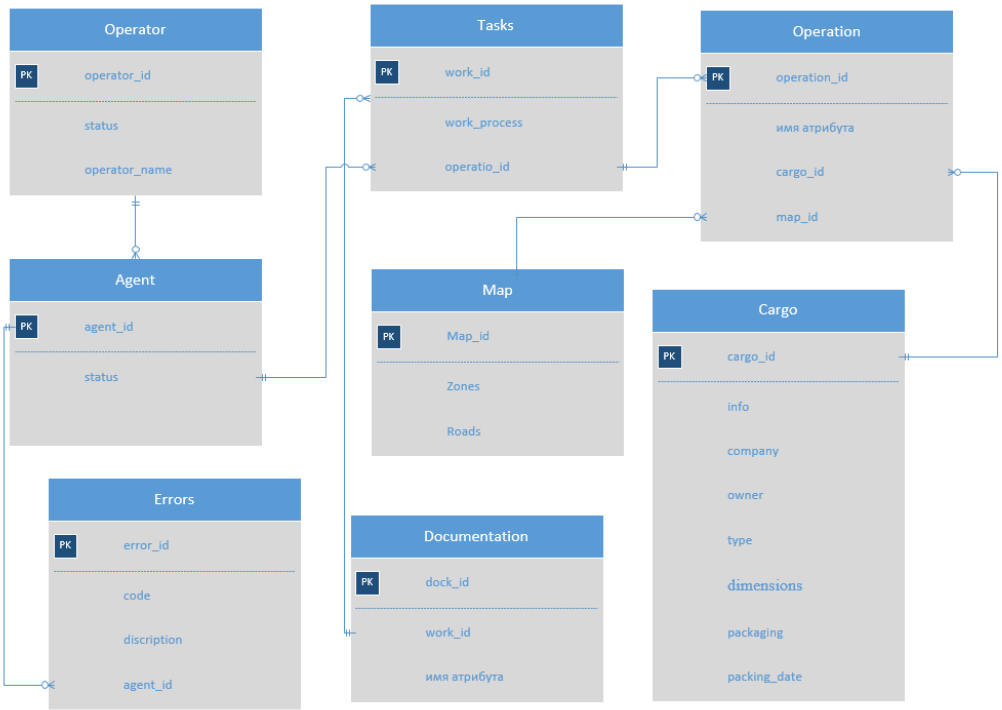

Рисунок 2.12. Схема бази даних

Таблиця Operator мітить відомості про людину яка слідкує за процесом роботи системи вона з’єднана з таблицею Agent зв’язком один до багатьох, оскільки в одного оператора може бути декілька агентів за якими він слідкує. Agent теж з’єднаний з таблицею Errors зв’язком один до багатьох, тому що у агенту може виникати декілька помилок одночасно. У агенту може бути декілька завдань які він повинен виконати, тому потрібно зв’язати таблиці зв’язком один до декількох. Tasks з’єднай з таблицею Operations для того, щоб мати список операцій які потрібно виконати над таблицею вантажів – Cargo яка з нею з’єднана, до списку Tasks під’єднана таблиця Map, в якій завжди є вся актуальна інформація потрібна для переміщення. У таблицю Documentation завжди поступає інформація про проведені роботи з таблиці Tasks.

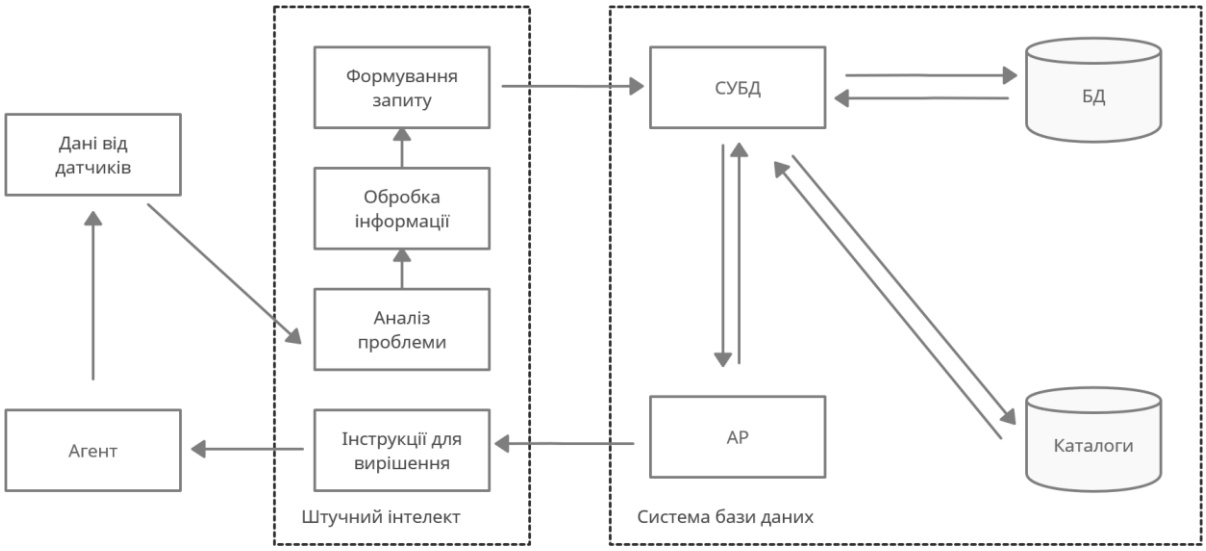

Рисунок 2.14. Схема зв'язків агенту, ШІ та СУБД

На схемі зв’язками агенту показано його взаємодію з ШІ та системою баз даних. Передача інформації від датчиків агенту до штучного інтелекту для визначення положення в просторі і потрібного алгоритму дій, які потрібно виконувати з вантажами, спочатку треба проаналізувати проблему, якщо проблема знаходиться в партерні типових, будь то пошук інформації про вантаж

по інформації з коду, система швидко виконує зчитування коду звертається до бази, щоб порівняти його з наявними якщо спів падінь немає потрібно виконати запит на додавання в базу. Чи не типових коли потрібно вирішити проблему з затором на дорозі, або пошук коду не дав результатів і не система не може сама знайти вирішення і потрібно відсилати повідомлення оператору що слідкує за правильністю роботи системи.

2.6 Код швидкого реагування

Система автопілоту буде тісно співпрацювати з базою даних, для внесення інформації про переміщення вантажів по території та їх вивозу з території промислового майданчику потрібно створити зручну і в той-же час потужну систему для ідентифікації вантажів, а також забезпечити мультиплатформність системи, щоб як агент так і людина, за допомогою певних апаратних та програмних застосунків мала можливість користуватись нею. Уявимо ситуацію в якій людині потрібно перевірити достовірність вантажів які знаходяться на складі або на склад прийшла велика кількість контейнерів з різним вмістом, потрібно всіх їх занести в базу, зазвичай це робиться в ручну та з використанням системи на основі QR кодів цей процес можна автоматизувати, і агент який однозначно буде оснащений як мінімум декількома камерами, легко зможе працювати з такими мітками. Під’їхавши до коробки яку потрібно перевести на певне місце на складі, він спочатку знайде її своїми візуальними сканерами, потім про сканує код який буде знаходиться з всіх сторін контейнера, точно визначить її вміст та зможе групувати її по місцях в які потрібно переміщувати певні вантажі. Люди зможе в ручному режимі знаходити місце в якому повинен буде розмішуватись вантаж за допомогою карти в якій по секторам буде позначене місце знаходження, а потім за допомогою камери на смартфоні або спеціального сканеру кодів під’єднаного до пк визначати точну коробку чи контейнере, наприклад для проведення перевірок, чи аудиту.

QR code - Quick Response Code, код швидкого реагування, це спеціальний тип матричних штрих кодів, які покликані для збереження інформації в виді мітки яку може зчитати машина з оптичним сенсором, яка зберігає інформацію про об’єкт до якого вона прив’язана. Може зберігати в собі більше 4 тисяч символів як цифр так і букв. В таку мітку можна помістити індикаційний номер вантажу або контейнеру і часткову інформацію про вантаж. Зчитавши такий код можна буде оперативно отримати всю необхідну інформацію, а також провести маніпуляції в базах даних. Використання кодів облегшить роботу не тільки для штучного інтелекту, а і для всього логістичного відділу підприємства, це дозволяє глобалізувати систему [8].

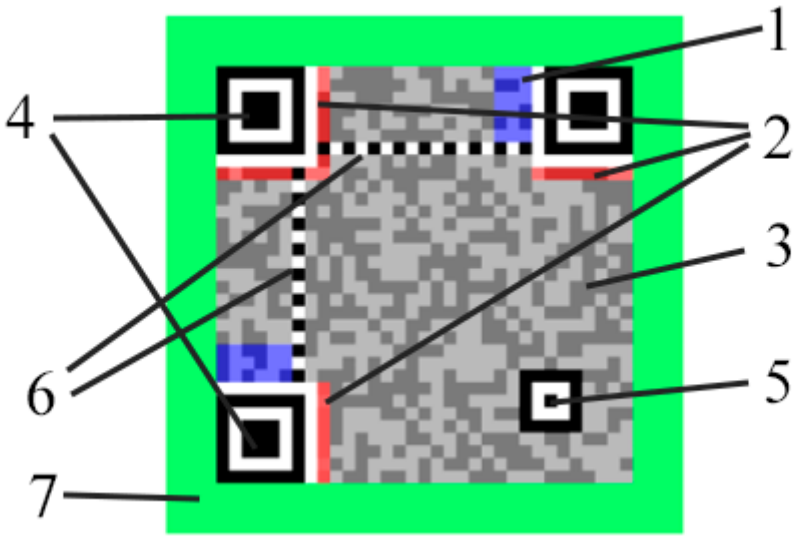

Рисунок 2.15 – Схема QR-коду

1. Синя зона, код версії – потрібен для визначення точної версії QR-коду;
2. Червона зона, код маски і рівня корекції – інформація в символах маскується через зміну модуля у визначених позиціях, щоб декодер розпізнав які з модулів потрібно зчитати, маски зберігаються в двох місцях разом з версією корекції помилок;
3. Сіра зона, данні з кодами корекції – ця частина коду зберігає в собі фактичні дані;

Незмінна частина QR-коду:

4. Пошуковий узор – визначає місцеположення і орієнтацію символу в полі зору;
5. Узор для вирівнювання – використовується для додаткової стабілізації коду, показує більш точну інформацію про його розміщення при розпізнаванні та, являє собою ключову мітку для декодування;
6. Лінії синхронізації – потрібні для визначення розмірів модулів коду;
7. Зелений колір, біле поле навкруги коду – важливий для програми інтервал сканування, потрібен, щоб відлічувати QR-код від його оточення;

Розпізнавання образів яке вбудовано в ШІ автопілоту легко навчити сканувати такі коди, а оскільки для зчитування кодів може підійти навіть мобільний телефон з самою простою камерою це дуже зручно для компаній. Тим більше, що така система забезпечую автоматичне ведення документації, завдяки тому ж алгоритму управління базами даних, це підвищує якість ведення документів і полегшує роботу співробітників компанії. Ці коди можна зробити нечитабельними для програм які не мають зв’язку з системами для програм які не є частинами системи, або навпаки зробити такі коди прозорими для глобалізації використання системи, адже стандартизація ведення документації може створити додатковий попит на технологію, що призведе до її здешевлення.

Рисунок 2.16 – QR-коду з зашифрованою в нього інформацією про вантаж.

Таблиця 1 – Зашифрована в код швидкого реагування інформація:

Ідентифікатор поля таблиці	Інформація поля:
Id:	00043431
Info:	032 - qr for shipping
Company:	everanthonycom
Owner:	everbrothercom
Type:	full size container
Dimensions:	w: 2m, h: 2m, l: 5m
Packaging:	moisture resistant
Packing date:	19.05.2021

Рисунок 2.17. Таблиця з даними зашифрованими в код швидкого реагування

В QR-код можна зашифрувати не менше чотирьох тисяч символів, тобто теоретично можна зашифрувати доволі великі кількість інформації, наприклад вміст багажу по пунктах. Загалом системи міток на основі матричних кодів зручні та прості у реалізації.

2.7 Система логістики переміщення вантажів по території порту.

Щоб створити середовище в якому агент під управлінням штучного інтелекту міг орієнтуватись в просторі потрібно створити мапу території з динамічно змінюючоюся позицією всіх вантажів які знаходяться в порту. Це допоможе вирішити зразу декілька проблем. Можна буде біль ефективно приймати, групувати, перемішувати вантажі по території порту. Ефективно вести декларації з внесенням всіх поступаючих і виїжаючих товарів. Підвищити ефективність роботи порту, позаяк використання паперу і неефективних таблиць значно сповільнює роботу працівників. А головною проблемою яку вирішить така

система є те, що можна буде вказуючи id вантажу сказати безпілотному навантажувачу перемістити його в нову точку базування або на панель відвантаження.

Використавши систему на основі камер можна вбудувати в неї алгоритм розпізнавання QR кодів нанесених на контейнери з вантажами, при чому коди зможуть сканувати за допомогою смартфонів і працівники компанії, що дозволить досягнути мультиплатформності вирішення. Звичайний працівник з смартфоном чи безпілотник з системою сканування кодів відскакувавши код зможе звернутись до системи і отримати інформацію про належність вантажу, його тип, його вміст, дату прибуття, місце відправки, власника, відправку і відгуку. Задавши запит до системи вона звертається до бази даних і надає інформацію яку запросив користувач, також можна буде в ручному режими задати якийсь перелік дій, які потрібно буде виконати з вантажем: переміщення в другу точку складу, переміщення в зону відвантаження, групування схожих вантажів, перепакування.

Рисунок. 2.18. Схема карти для переміщення агенту

Зелені зони це зони в яких агент повинен забирати вантажі

Жовті лінії, це сітка маршрутів по яким може рухатись ШІ.

Оранжева область, це зона відвантаження контейнерів.

Червона зона, зона для паркування і очікування агенту.

Чорна лінія, це оптимальний маршрут з точки А в Б.

Синя зона, це територія підприємства.

Алгоритм знаходження найкоротшого з можливих шляхів використовує досить прості правила у моделюванні системи, використовується вже знайомий нам з університетської програми алгоритм Дейкстри, для знаходження найкоротших шляхів в зваженому графі. Фактично нам потрібно визначити вершини графу зони в яких наш агент виконує роботу та назначити кожному відрізку дороги свою вагу.

Рисунок 2.19. мапа об’єкту з вписаним в неї графом пошуку найкоротшого шляху

При надаванні ваг ребрам графу потрібно використовувати фізичну мапу місцевості і відштовхуючись від дистанції в сотнях метрів або в кілометрі надавати вагу ребру.

Граф можна записати у вигляді матриці Табл.2 :

Таблиця 2 – Матриця обходу графу за допомогою алгоритму Дейкстри:

	1	**2**	**3**	**4**	**5**
1		4	10		3
2	4		5	6	5
3	10	5		7	
4		6	7		8
5	3	5		8	

Рисунок 2.20. структура матриці обходу графу

Припустимо навантажувач забрав контейнер з зони відвантаження яка знаходиться під вершиною графу 4 і потрібно перемістити вантаж в зону під вершиною 1.

Даний алгоритм по крокам перебирає усі вершини графу, і назначує їм мітки, які являються відомими мінімальними відстанями від стартової вершини до потрібної вершини.

З вершини джерела прокладемо найкоротший маршрту до вершини один, оскільки немає маршруту без вершин посередників проводимо лінії до них і переходимо до наступного кроку. Кожній з розглянутих вершин призначимо мітку дорівнює сумі мітки W і довгі шляхи з W в розглянуту вершину, але тільки в тому випадку, якщо отримана сума буде менше попереднього значення мітки. Якщо ж сума не буде менше, то залишаємо попередню позначку без змін [7].

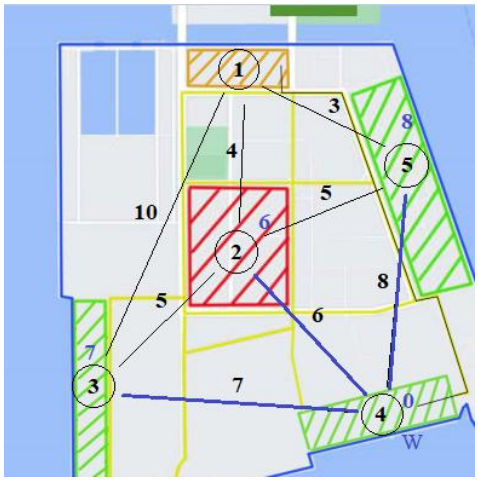

Рисунок 2.20. початок обходу графу

Далі додаємо до відмічених ребер, ребра які закінчують маршрут до вершини.

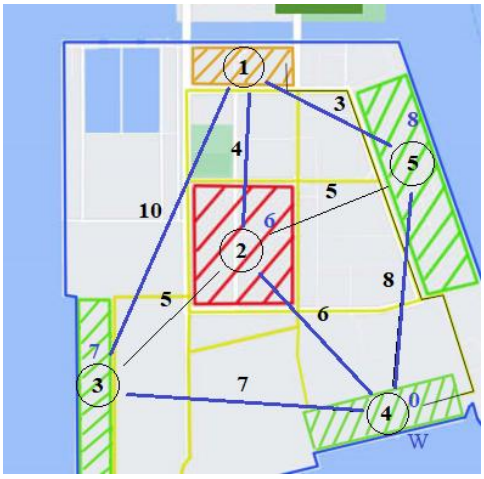

Рисунок 2.21. заключний етап обходу

Той маршрут, сума вагів ребер якого буде найменша і являє собою оптимальний маршрут.

1. (4, 2, 1) = 6 + 4 =10
2. (4, 5, 1) = 8 + 3 = 11
3. (5, 3, 10) = 7 + 10 = 17

Оптимальний шлях: (4, 2, 1).

Таким чином, методом виключення ребер графу можна виключати і маршрути прокладені по реальним дорогам комплексу, які прив’язані до таких ребер, пошук шляху до зони яка позначена однією з вершин графу потрібно починати з виключеними ребром яке означає заблокований шлях.

Рисунок 2.22. Мапа зі знайденим альтернативним шляхом

Для забезпечення нормальної роботи системи спочатку потрібно створити карту в якій безпілотний навантажувач зможе нормально рухатись, позначивши дороги по яким зможе рухатись автопілот, позначити на ній зони відвантаження і зберігання контейнерів або вантажів інших габаритів, позначити зони де вантажі будуть відвантажуватись з кораблів та зони в яких вони буду грузитись на фури для відправки. Точно визначивши маршрути по яким зможе рухатись безпілотний навантажувач можна буде виріши проблеми з орієнтуванням в просторі, так як орієнтуватись техніка буде по вже заздалегідь визначеним маршрутам які вже внесені в карту системи. Досить буде указати точку в якій знаходиться система і точку в яку їй потрібно прибути та прорахувати оптимальний маршрут

слідування. За необхідності таку карту можна буде змінювати в режимі реального часу, досить буде лише зайти в графічний інтерфейс системи і визначити нові зони проїзду.

Створена карта буде завантажена в базу даних та буде використовуватись при виконанні агентом роботи, в зона де потрібно буде брати вантажі агент повинен буде знайти і розпізнати його, потім він повинен прочитати його QR код, який буде розмішений на його на його сторонах, прочитавши код він отримає його індикаційний номер зробивши запит до системи управління отримає ряд дій які він повинен виконати з вантажем, наприклад точку куди його потрібно перемістити, потім йому потрібно буде розпізнати мітки з точками підвісу для того щоб підняти контейнер, потім потрібно буде розміститись в зручному положенні для виконання підйому і підгнити вантаж, далі виконання дій по перевезенню вантажу в задану точку орієнтуючись по мапі місцевості яку ми заздалегідь створили, і вкінці знайти точку в якій можна розмістити вантаж і занести в базу нову точку дислокації, виконавши роботу агент повинен взяти зі списку наступну роботу по пре мішенню і перейти до її виконання.

Зона складу де можуть зберігатись різні вантажі повинна бути розділена на сектори зберігання які будуть маркуватись як матриці, зверху латинськими літерами від A до Z, з боку арабськими цифрами від 1 і до скільки буде необхідно. Сектори необхідні для більш зручного пошуку потрібних вантажів чи контейнерів, оскільки зона складу може бути гігантською і позначити на ній точні координати кожного вантажу окремо буде не можливо і мало ефективно. Буде велике нагромадження моток на одному місці мапи. А так можна вказати що вантаж знаходиться на 3 ярусі контейнерів зверху, в зоні складу 1, сектор C3. Транспорт який під’їде забрати вантаж відсканує код і забере його. Це досить проста реалізація сортування та розмежування території складу, та навіть якщо просто розділити його на велику кількість простих секторів це вже у декілька розвів збільшити продуктивність роботи складу, оскільки ведення нумерації без прямого доступу до бази даних не дає якогось значимого переваги, а з доступом в базу для агенту і з доступом до карти з розмежуванням для персоналу в режимі

реального часу дає змогу знаходити вантажі в декілька разів швидше. Єдиною дійсно значною проблемою являється різний форм-фактор різних видів вантажів, та можна просто помішати їх в зони з специфічним рамками, де вони будуть розташовані у відносно хаотичному порядку, але все ще будуть досить легко знаходитися за допомогою мапи та бази даних.

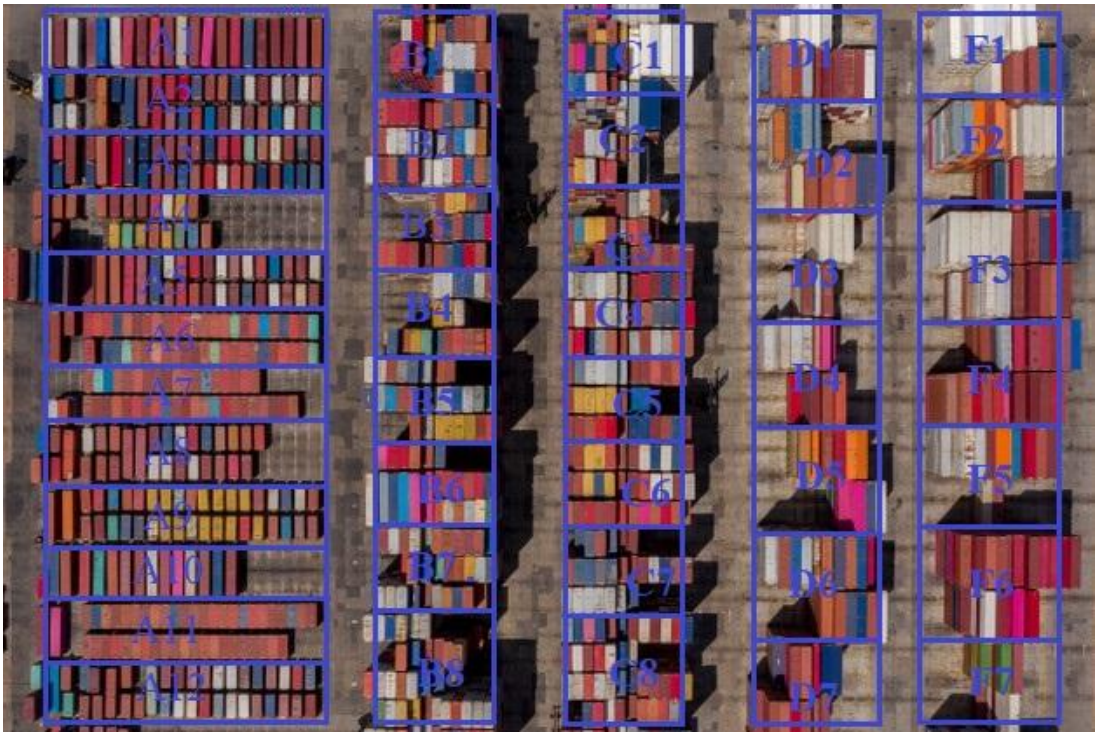

Рисунок 2.23. зона з позначеними на ній секторами

Висновок до другого розділу:

В другому розділі ми розглянули якими технологіями і концептуальними рішеннями можна послуговуватись для створення систем з штучним інтелектом доля автоматизації роботи промисловості. Розглянули за рахунок яких систем ШІ бачить навколишній світ, ознайомилися з методикою створення кодів швидкого реагування з використанням якої створює зрозумілий і простій в користуванні інтерфейс об’єкт-машина, об’єкт-база даних., розібрали які алгоритми потрібно використовувати для створення карт і пошуку оптимального маршруту. Визначились з алгоритмом і особливостями роботи база даних, створили алгоритм роботи з базою та її структурну схему.

3 ОПИС ІНФРАСТРУКТУРИ ПРОГРАМНОГО ЗАБЕСПЕЧЕННЯ

3.1 Інфраструктура програми

Для створення системи знадобиться виготовити навантажувачі які будуть оснащені системами управління, усіма необхідними датчиками та матимуть на борту досить потужне апаратне забезпечення, для того, щоб мати змогу опрацьовувати увесь той потік інформації який буде поступати від датчиків або модернізувати вже наявні на підприємстві одиниці техніки, такі як навантажувачі, обладнання їх датчиками всього половина роботи, потрібно вносити зміни в систему управління, додавати механізми повороту керма та перемикання передач, маніпулювання пристроєм захоплення та піднімання вантажу, що може являти собою серйозну проблему навіть на етапі проектування, а на етапі реалізації може стати взагалі не можливим на практиці. Тому створення транспорту який з самого початку призначений для виконання таких робіт є більш доцільним. Агент можна оснастити двома камерами які дивляться в перед, однією широкоформатною риб’ячим оком для того щоб мати широкий кут огляду з переду і з вузьким фокусуванням для впевненого переміщення на високій швидкості, боковими камерами розміщеними з переду під кутом в 30 градусів для гарного огляду з сторін, для того щоб не зачіпляти ненароком об’єкти по бокам та задня камера для здавання назад і можливо якихось специфічних маніпуляцій. В якості обчислювальної машини буде виступати система на базі intel core i7-9700k, процесор має вісім фізичних потоків, томі від прекрасно підходить під реалізацію розбиття роботи мережі на декілька, в даному випадку вісім окремих процесів, що з використанням openCV, яка якраз і допомагає реалізувати дану процедуру, суттєво підіймає продуктивність роботи системи. Максимальна тактова чистота процесору 4.9 ГГц, має 2 канали для пам’яті, підтримує пам’ять DDR4 до 64гб одночасно, був створений під socket 1151, підтримує Turbo Boost. Відео карта відіграє найважливішу роль в роботі з візуальними зображеннями та відео

потоками, особливо коли діло йде про розпізнавання образів, з використанням технології NVIDIA CUDA можна в сотні разів прискорити обробку, що скоротить час розпізнавання з 10 секунд до декількох сотень мілісекунд. Тому в даній задачі ми будемо використовувати NVIDIA 2070 – це потужна і вже перевірена у використанні з YOLO видеокарта і її продуктивність можна спрогнозувати. Все це буде зібрано на базі материнської плати Asus PRIME B365-PLUS, проста, дешева і надійна плата, яка повністю підходить для виконання поставлених перед нею задач, підтримує до 64 ГБ пам’яті одночасно, що є плюсом. Пам’яті буде встановлено 64 ГБ DDR4, виробник особливої ролі грати не буде. Також потрібно щоб в системі вистачало місця для збереження інформації, так можна встановити ssd Samsung на 512 Гб, та HDD Western Digital на 1 Тб, на SSD буде записана база з вагами для розпізнавання оскільки ця інформація постійно знаходиться в роботі з системою і тому швидкість відповіді на запити до бази з вагами суттєво підвищує швидкість роботи усієї системи, також на ssd буде записана операційна система і увесь необхідний софт, на HDD буде записуватись документація оскільки вона буде поступово передаватись на сервер, а потім завантажуватись в базу, щоб уникнути так званого bottleneck, коли по wifi не можна буде зразу миттєво передати велику кількість інформації, спочатку вона буде записана на жорсткий диск, потім поступово шматками завантажуватись на сервер, а звідти вже в базу. Живлення потрібно організувати від системи самого транспортного засобу, це не така вже і складна проблема, адже більшості комплектуючим достатньо живлення напругою від 12 до 24 вольт, тому модифікувавши блок живлення комп’ютера можна добитися бажаного результату, можна використати як зразок ThermalTake 650W з срібним сертифікатом в вісімдесят балів, надійний і доволі не дорогий блок живлення.

Для створення ефективної системи потрібно орендувати виділений хмарний сервер, на якому буде розгорнута база даних з інформацією про вантажі і мапою території, на такому сервері повинне бути резервне копіювання інформації, так як якщо при пошкодженні або виходу із ладу одного диску з інформацією вийде з ладу в ся база даних, це потягне за собою зупинку роботи

системи, а вже за цим по слідують великі збитки для компанії. Можна зупинитись на Microsoft Azure, так як вона підтримує платформу розробки .NET яка являється похідною від мови програмування C# розробленою все тією ж компанією Microsoft. Має сервіси зберігання REST API та SDK API, для збереження даних в хмарі та доступу до них. Table Service дозволяє програмам зберігати структурований текст в секціонований колекції сутностей, доступ до яких здійснюється по ключу секціонування і первинному ключу.

Система написана на C# з використанням Entity Framework для побудови баз даних, для створення веб частини контролера системи був використаний ASP.NET за допомогою цієї технології на C# можна будувати веб сайти, писати код для платформи ASP.NET можна практично на будь-яких мовах, включно з C#. Компіляція коду на ньому виконується швидка і до того ж більшість похибок знаходяться на етапі розробки. Вся система буде функціонувати середі .Net Framework.

Зв’язок з агентами буде бездротовий і поширюватись за допомогою вишок розставлених по території транслюючись у вигляді wifi хвиль, це проста у використанні і досить дешева технологія, використовуючи гарні передавачі wifi сигналу можна добитися максимального покриття території навіть від декількох десятків вишок, до того ж по wifi можна зв’язати прилади які є на руках у персоналу, такі як: телефони, планшети, їх не потрібно буде модернізувати перед використанням достатньо буде закупити звичайні моделі і встановити на них програму для зчитування кодів яку можна розробити спеціально для компанії, програму для ведення документації і мапу з розмещенням усіх вантажів по території а по вайфай зв’язати пристрій з хмарним сервісом системи де знаходиться основна база даних. Таке рішення вирішує зразу декілька проблем, є супер простим у реалізації і на останок досить дешевим.

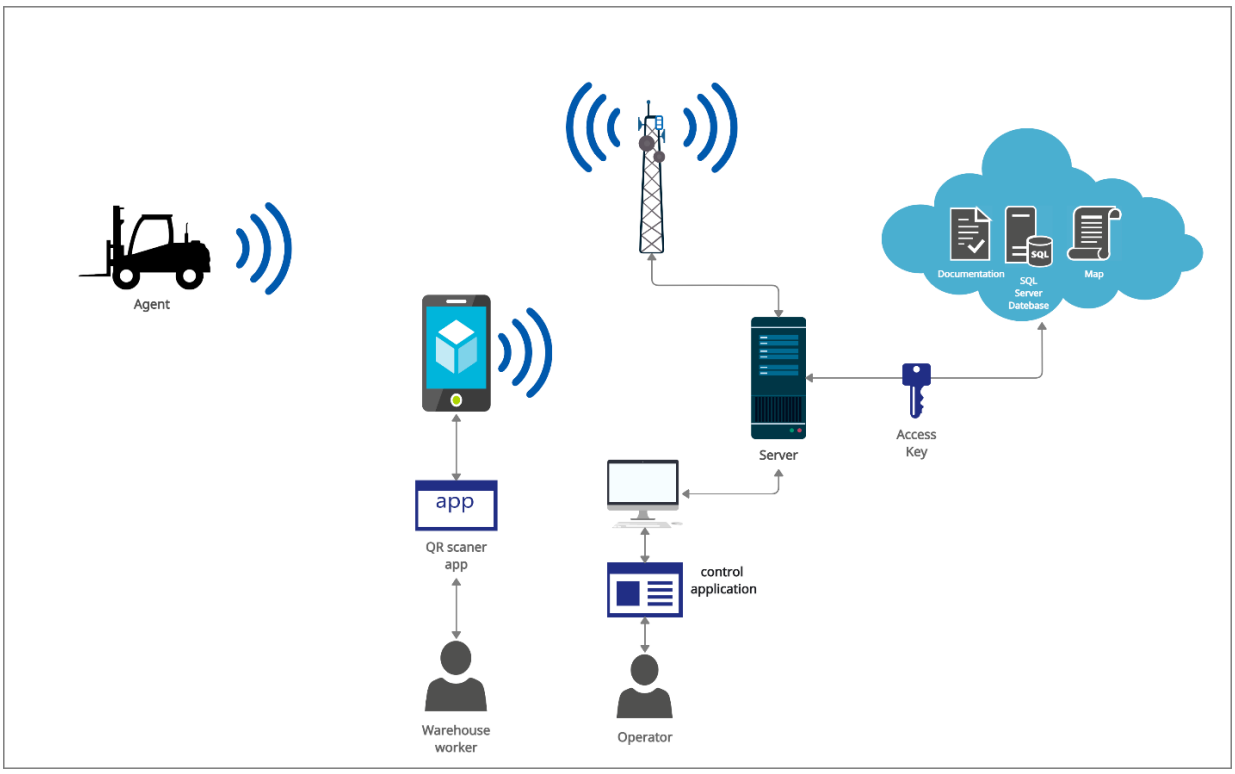

Рисунок 3.1. Схема роботи алгоритму

На схемі зображене використання системи з використанням хмарного сервісу, можливе розміщення такого серверу і фізично на території підприємства, та зазвичай використання хмарного сервісу зручніше як для малого підприємства оскільки такі рішення зазвичай дешевші, так і для великої компанії яка має під свої контролем декілька складів або ж морських портів. Оскільки забезпечує як централізоване збереження інформації та резервне копіювання так і дозволяє використовувати один сервер для багатьох окремих систем. На цій схемі документація, SQL сервер з базою даних та мапа зберігається в хмарі, можна також створити локальний сервер на якому може зберігатись мапа території та база з даними про вантажі а в хмарі буде зберігатись тільки документація, така система буде менш залежна від доступу в інтернет, і її база буде біль захищена від хакерів. Зв’язок сервера по збереженню і обробці інформації з хмарним сервером проходить по зашифрованому протоколі з використанням ключів доступу, Обмін інформацією та взаємодія з агентом проходить по мережі wifi, також по цій мережі проходить взаємодія і з портативними пристроями працівників складу, вони взаємодіють з системою та її базою даних за допомогою спеціального

програмного забезпечення, яке покликане облегшити їх роботу та позбавити їх паперової роботи.

На рівні агенту інформація з датчиків, а саме відеоряд з камер буде передаватися до системи управління агенту де нейронна мережа YOLO буде опрацьовувати вхідний відеоряд і передавати його до системи штучного інтелекту, де на основі розпізнаних образів з відеоряду буде проводитись процес штучного мислення і будуть прийматись рішення, далі рішення будуть переводитись в зрозумілі для апаратного забезпечення сигнали, агент буде приводитись в рух і виконувати поставленні перед ним завдання. Взявши конкретно завдавання по перевезенню вантажу з місця на місце ШІ потрібно вирішити декілька задач, визначити чи потрібно переміститись в положення вантажу, дізнатись де конкретно зараз він знаходиться, забрати його в місці де він знаходиться, визначити точку в яку його потрібно перемістити, відкрити карту і дізнатись як він може туди добратись, при цьому він повинен виконувати завдання які він може брати з списку, якщо завдання закінчились він повинен зупинити діяльність і повернутись в точку базування, а при знаходженні нового завдання знову почати виконувати роботу.

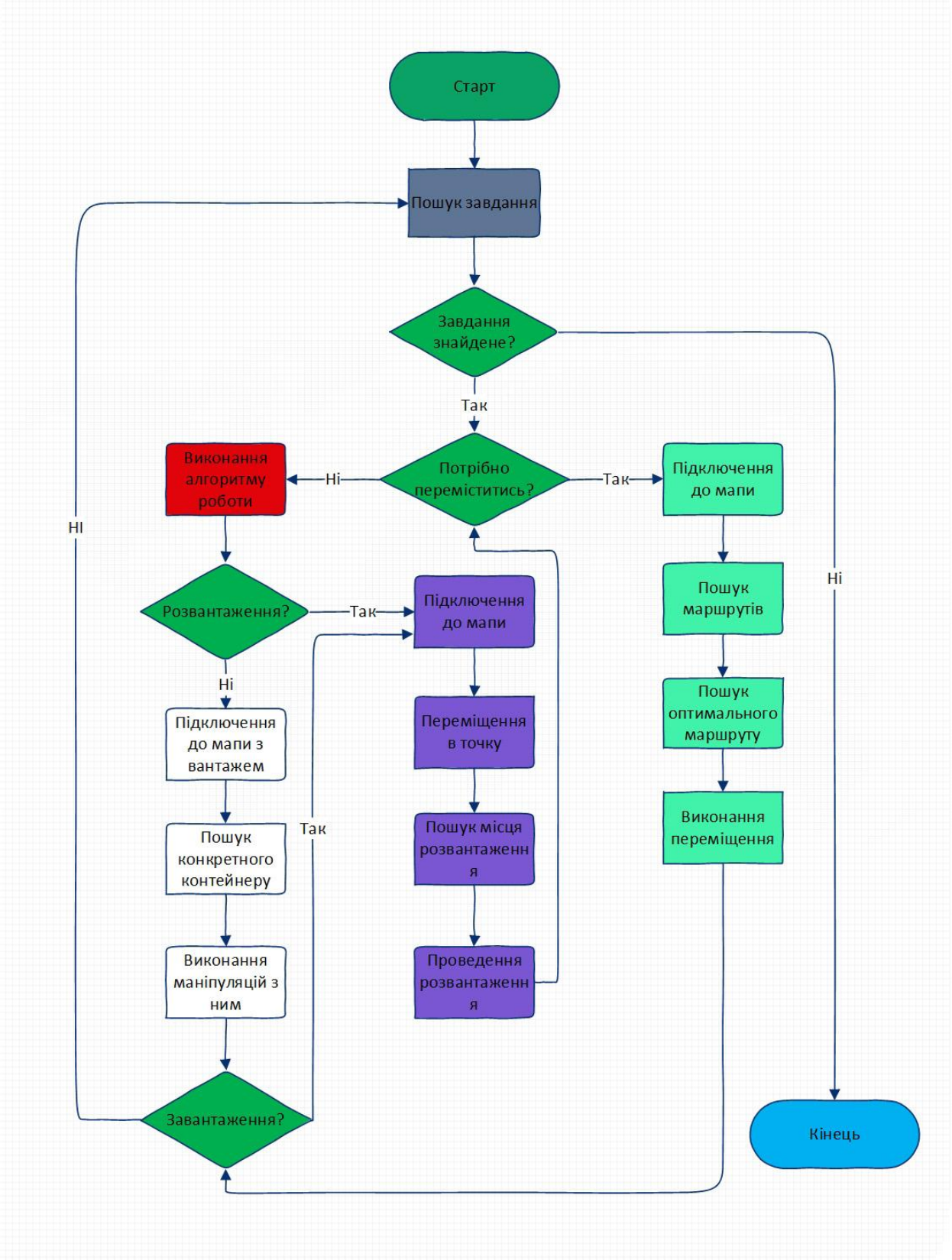

Рисунок 3.2. Блок-схема роботи алгоритму переміщення вантажів.

3.2 Система розпізнавання образів YOLOv4

YOLO – це надзвичайно точна, працююча в реальному часі нейронна сітка для розпізнавання образів. Real time нейронна мережа повинна обробляти відео високої якості в розрішенні 1920 на 1080 на льоту в фреймрейті мінімум 60 кадрів за хвилину. У більшості практичних завдань є мінімально необхідні вимоги до детекторів - це мінімально допустимі точність і швидкість. Зазвичай мінімально допустима швидкість від 30 FPS (кадрів в секунду) і вище для real-time систем.

З коробки вона вже несе в собі патерни вагів для розпізнавання великої кількості предметів на зображеннях, ось деякі із них: люди, транспорт, вантажівки. Вона розповсюджується в якості безкоштовного програмного забезпечення з відкритим кодом, тому ми можемо її використовувати для тестування роботи системи, а за необхідності використання її в комерційних цілях нам просто потрібно буде поговорити з розробниками за ліцензування модифікованої версії яку ми розробимо, адже для більш вузько направленого використання її для розпізнавання образів на складі потрібно буде натренувати нейронну сітку на розпізнавання специфічних об’єктів таких як контейнери, навантажувальна техніка, особливі маркування, додати в неї функцію розпізнавання QR кодів.

Великою проблемою в створенні систем для розпізнавання образів, є необхідність використання дорогого обладнання, яке використовує велику кількість електроенергії виділяє велику кількість тепла займає багато місця своїми габаритами, оскільки для використання систем розпізнавання образів необхідно мати велику обчислювальну потужність. Тому оптимізація роботи таких систем являє ключову проблему, для нейронної мережі YOLOv4 представлено декілька засобів для оптимізації її роботи. Для нормальної роботи мережі використовують framework darknet і декілька засобів оптимізації openCV та cuda.

OpenCV (open Source Computer Vision Library), бібліотека комп’ютерного бачення з відкритим вихідним кодом, бібліотека була розроблена для ефективного використання ресурсів процесорів лінії intel, для підвищення конкурентоспроможності платформи intel для розробників галузі комп’ютерного бачення, за рахунок використання intel Performance Libraries, низькорівневих

бібліотек для обробки сигналів, зображень, а також медіа кодеків та MKL. Власне OpenCV і здатна застосовувати багато поточність процесорів, за рахунок чого різко росте продуктивність системи.

CUDA - це програмно-апаратна архітектура паралельних обчислень, що дозволяє істотно збільшити обчислювальну продуктивність завдяки використанню графічних процесорів NVIDIA. CUDA дозволяє програмістам реалізовувати на спеціальному спрощеному діалекті мови C алгоритми, які використовуються в графічних процесорах NVIDIA, і включати спеціальні функції в текст програми на C [4].

Модель Yolo була розроблена для нейронної мережі на основі DarkNet, для нас же деякі особливості цього рішення не підходять. DarkNet зберігає навчені коефіцієнти (ваги) в форматі, який може бути розпізнаний за допомогою різних методів на різних платформах. Ця проблема може бути каменем спотикання, тому що вам може знадобитися навчити модель на надпотужний обладнанні, а потім використовувати її на іншому обладнання. DarkNet написаний на C і не має іншого програмного інтерфейсу, тому, якщо вимоги платформи або власні переваги змусять вас звернутися до іншої мови програмування, вам доведеться додатково попрацювати над його інтеграцією. Також він поширюється тільки в форматі вихідного коду, і процес компіляції на деяких платформах може бути досить проблематичним. YOLOv4 вимагає в 5 рази дешевше обладнання і при цьому точніше, ніж EfficientDet-D2 (Google-TensorFlow). Можна використовувати EfficientDet-D0 (Google-TensorFlow) тоді вартість обладнання буде однакова, але точність буде на 10% AP нижче. Крім того, деякі промислові системи мають обмеження по тепловиділенню або по використанню пасивної системи охолодження - в цьому випадку ви не зможете використовувати TitanV навіть маючи гроші.

Рисунок 3.3. Результат розпізнавання образів за допомогою YOLO

При використанні YOLOv4 (416x416) на GPU RTX 2080 Ti з використанням TensorRT + tkDNN ми досягаємо швидкість в 2x рази більше, а при використанні batch = 4 в 3x-4x рази більше - для чесного порівняння ми не наводимо ці результати в статті на arxiv. org:

Size	Darknet FPS (avg)	tkDNN TensorRT FP32 FPS	tkDNN TensorRT FP16 FPS	OpenCV FP16 FPS	tkDNN TensorRT FP16 batch=4 FPS	OpenCV FP16 batch=4 FPS	tkDNN Speedup
320	100	116	**202**	171	**423**	384	4.2x
416	82	103	**162**	146	**284**	260	3.5x
512	69	91	**134**	125	**206**	190	2.9x
608	53	62	**103**	100	**150**	133	2.8x

nVidia GPU GeForce RTX 2080 Ti

Рисунок 3.4. Порівняння роботи Yolo на різних фреймворках

Відкритий фреймворк для прикладного використання згорточної нейронної мережі, що підходить, наприклад, для класифікації фотографій або виділення об'єктів на зображеннях в режимі реального часу. Код написаний на Ci з використанням CUDA. Обчислення можуть проводитися на CPU і GPU.

Самостійне навчання (SAT) - це техніка збільшення даних. Спочатку він виконує прямий прохід на навчальному зразку. Традиційно під час зворотного розповсюдження ми регулюємо ваги моделі, щоб покращити детектор при виявленні об’єктів на цьому зображенні. Тут справа йде у зворотному напрямку. Це змінює зображення таким чином, що може найбільше погіршити продуктивність детектора. тобто створює змагальну атаку, націлену на поточну модель, навіть якщо нове зображення може виглядати візуально однаково. Далі модель навчається з цим новим зображенням з оригінальним межовим полем та міткою класу. Це допомагає узагальнити модель та зменшити переважання [10].

Для випадку $b_x = c_x$ та $b_x = c_x + 1$ нам потрібно, щоб t_x мав величезне негативне та позитивне значення відповідно. Але ми можемо помножити σ на коефіцієнт масштабування (> 1,0), щоб полегшити. Ось зміни вихідного коду:

2. Eliminate grid sensitivity - by using `scale_x_y=` parameter from `[yolo]` layer in cfg-file (by default use `scale_x_y=1.0` if the value is not set in cfg file)
we should use:

```
const float x_tmp = logistic_activate(srcData[box_index + 0]) * scale_x_y - (scale_x_y - 1) / 2;
const float y_tmp = logistic_activate(srcData[box_index + 1]) * scale_x_y - (scale_x_y - 1) / 2;
dstData[box_index + 0] = (x + x_tmp)) / cols;
dstData[box_index + 1] = (y + y_tmp)) / rows;
```

instead of

opencv/modules/dnn/src/layers/region_layer.cpp
Lines 305 to 306 in 51a42c0

```
305     dstData[box_index + 0] = (x + logistic_activate(srcData[box_index + 0])) / cols;
306     dstData[box_index + 1] = (y + logistic_activate(srcData[box_index + 1])) / rows;
```

Рисунок 3.5. Вивід в консоль інформації про роботу

Завдяки простоті реалізації роботи нейронної мережі, а також завдяки широкому спектру програмного забезпечення для оптимізації роботи такої нейронної мережі, YOLO являє собою простий варіант для розробки системи автопілоту. Реалізована система самостійного навчання також відіграє важливу

функцію в виборі саме такого типу нейронної мережі, так як на модифікацію функціоналу буде витрачатись значно менша частина часу, на відміну від звичайних нейронних мереж, велику роль також відіграє системи ділення зображення на частини та системи по оптимізації розрахунків в середині нейронної мережі, також гані результати на тестах по точності розпізнавання, коли деякі параметри показували 95-99% на точну відповідь.

3.3 СУБД ТА БАЗА ДАНИХ

Entity Framework являє спеціальну об'єктно-орієнтовану технологію на базі фреймворка .NET для роботи з даними. Якщо традиційні засоби ADO.NET дозволяють створювати підключення, команди та інші об'єкти для взаємодії з базами даних, то Entity Framework являє собою більш високий рівень абстракції, який дозволяє абстрагуватися від самої бази даних і працювати з даними незалежно від типу сховища. Ми вже працємо з об'єктами Entity Framework, індексами, первинними і зовнішніми ключами, на концептуальному рівні, який нам пропонує Entity Framework. Entity Framework - 1.0 вийшла ще в 2008 році і мала дуже обмежену функціональність, базову підтримку ORM (об'єктно-реляційне зіставлення - відображення даних на реальні об'єкти) і один єдиний підхід до взаємодії з бд - Database First. З виходом версії 4.0 у 2010 році багато чого змінилося - з цього часу Entity Framework стала рекомендованої технологією для доступу до даних, а в сам фреймворк були введені нові можливості з бд - підходи Model First і Code First. Додаткові поліпшення функціоналу пішли з виходом версії 5.0 в 2012 році. І нарешті, в 2013 році був випущений Entity Framework 6.0, що володіє асинхронним доступом до даних. Центральної концепцією Entity Framework є поняття сутності або entity. Сутність представляє набір даних, асоційованих з певним об'єктом. Тому дана технологія передбачає роботу не з таблицями, а з об'єктами і їх наборами. Будь-яка сутність, як і будь-який об'єкт з реального світу, має низку властивостей. Наприклад, якщо сутність описує людини, то ми можемо виділити такі властивості, як ім'я, прізвище, зріст,

вік, вага. Властивості необов'язково представляють прості дані типу int, а й можуть представляти більш комплексні структури даних. І у кожної сутності може бути одна або кілька властивостей, які будуть відрізняти цю сутність від інших і будуть унікально визначати цю сутність. Подібні властивості називають ключами. При цьому суті можуть бути пов'язані асоціативної зв'язком один-ко-многим, один-ко-одному і багато-до-багатьох, подібно до того, як в реальній базі даних відбувається зв'язок через зовнішні ключі. Відмінною рисою Entity Framework є використання запитів LINQ для вибірки даних з БД. За допомогою LINQ ми можемо не тільки отримувати певні рядки, що зберігають об'єкти, з бд, а й отримувати об'єкти, пов'язані різними асоціативними зв'язками [8].

База даних складається з семи таблиць, Operator з таблицею Agent зв'язаний один до кількох, адже один оператор може слідкувати одночасно за декількома агентами під управління ШІ, він створений за допомогою Entity Framework – це вирішення для управління базами даних сімейства мов .NET, воно дозволяє взаємодіяти з СУБД за допомогою сутностей entity, а не таблиць. Тому код з використанням NF пишеться набагато швидше.

```
namespace DatabaseHelper
{
    public class Operator
    {
        public int operator_id { get; set; }
        public string status { get; set; }
        public string operator_name { get; set; }
        public int role_id { get; set; }
    }

}
```

Рисунок 3.6. Модель таблиці даних Operator

При створенні баз даних на основі EF обов’язкова присутність первинного ключу, в даній таблиці це поле id. Перед тим як оператор приступить до роботи, його потрібно зареєструвати в системі та видати йому роль згідно його посади, де 0 – це рівень доступу молодший спеціаліст, який може тільки слідкувати за роботою системи, 1 – це спеціаліст, який слідкує за роботою системи та може

вмішуватись в її роботу в обмеженому секторі прав згідно своєї ролі, 2 – це рівень доступу, старший спеціаліст який може брати на себе керування в системі в складних ситуаціях, 3 – це рівень доступу майстер який може під свою відповідальність надавати розширені права доступу другим членам системи коли це необхідно в складних ситуаціях. Статус це поле в мережі оператор чи ні.

Як тільки оператор сідає за пульт керування йому видає підлеглих, якщо його статус роль третього рівня, або якщо його роль нижче третього рівня йому видає агенти роботу яких він буде корегувати. Таблиця агент має декілька полів, поле ідентифікатору agent_id, поле типу техніки type_id, яке означає конкретний тип техніки і роботи яку може виконувати така техніка, це потрібно для правильного направлення техніки на роботу яка її підходить, адже не можна відправити вантажівку на розвантаження контейнерів з машини, а от привезти її забрати вантаж можна, поле статус яке означає статус техніки в даний момент, зайнята вона чи ні.

```
namespace DatabaseHelper
{
    public class Agent
    {
        public int agent_id { get; set; }
        public int type_id { get; set; }
        public string status { get; set; }

    }
}
```

Рисунок 3.7. Модель таблиці даних Agent

При виконанні роботи завжди продукується велика кількість помилок, від банального від’єднання від мережі, до серйозних системних помилок переповненості буферів пам’яті або неможливості виконання агентом завдання. В таких ситуація записування та документування помилок, являє не тільки гарний інструмент для їх вирішення але і прекрасний інструмент для збору статистичної

інформації і подальшого вдосконалення системи. Тому в таблицю errors входять такі поля: error_id це звичайний ідентифікатор, code –, , description – короткий опис, тут записано, що помилка собою означає, agent_id – це агент від якого поступила помилка, для точної ідентифікації машини.

```
namespace DatabaseHelper
{
    public class Errors
    {
        public int error_id { get; set; }
        public int code { get; set; }
        public string discription { get; set; }
        public int agent_id { get; set; }
    }
}
```

Рисунок 3.8. Модель таблиці даних Errors

Для выкованная робіт агенту потрібна таблиця з списком завдань, в таку таблицю будуть потрапляти завдання у вигляді черги, де додавання нового завдання йде в кінець черги, а береться завдання з початку. У таблиці буде ідентифікатор по якому можна буде точно відслідковувати завдання, поле work_process яке означає статус виконання процесу на, agent_id означає ідентифікатор конкретного агенту, який виконує завдання, таблиця Tasks зв’язана з таблицею Map зв’язком один до одного, оскільки логістика вантажів по території передбачає роботу з мапою і позиціями вантажів на ній.

```
namespace DatabaseHelper
{
    public class Tasks
    {
        public int work_id { get; set; }
        public string work_process { get; set; }
        public int operator_id { get; set; }
        public int cargo_id
    }
}
```

Рисунок 3.9 Модель таблиці даних Tasks

В таблицю даних Cargo потрапляє уся інформація про вантаж, у поле cargo_id записаний унікальний ідентифікатор вантажу, цей ідентифікатор записується також як головне поле QR коду, тобто це головна стрічка інформації в таблиці вантажу, так як по ній і здійснюється його логістика по території. Поле info містить інформацію про вантаж, які матеріали знаходяться в контейнері чи коробці, тип продукції. В поле company записана інформація про компанію якій цей вантаж належить. В поле owner записана інформація про компанію яка на даний момент зберігає вантаж. Type відповідає як за конкретний тип вантажу так і за спосіб його пакування, чи то габаритний контейнер чи проста коробка. Поле dimensions – це габаритні розміри контейнеру, загалом суто його фізичні характеристики які іноді потрібні для проведення точних маніпуляцій: завантаження, відвантаження, перевезення. Packer_id – це ідентифікатор людина яка проводила перевірку вантажу і запакувала його поставивши пломбу. Packing_date – це дата коли контейнер або упаковка коробки була запечатана і опломбована з попередньою перевіркою вмісту.

```
namespace DatabaseHelper
{
    public class Cargo
    {
        public int cargo_id { get; set; }
        public string info { get; set; }
        public string company { get; set; }
        public string owner { get; set; }
        public string type { get; set; }
        public int dimensions { get; set; }
        public int packer_id { get; set; }
        public Nullable<System.DateTime> packing_date {get; set; }
    }
}
```

Рисунок 3.10. Модель таблиці даних Cargo

В таблицю Documentation записується інформація ідентифікатор документу – dock-id, date поле в яке записується дата створення документу, work_id ідентифікатор роботи при виконанні якої був створений цей документ для того щоб зв’язувати між собою інформацію про документ та роботу, agent_id – це ідентифікатор агенту який проводив виконання роботи, description – опис,

створюється для легкого та стислого викладення інформації в документі, для підвищення простоти читання з сторони людини, content – частина з основним вмістом документу, туди записується вся потрібна при документуванні інформація.

```
namespace DatabaseHelper
{
    public class Documentation
    {
        public int dock_id { get; set; }
        public Nullable<System.DateTime> date { get; set; }
        public string work_id { get; set; }
        public string agent_id { get; set; }
        public string description { get; set; }
        public string content { get; set; }
    }
}
```

Рисунок 3.11. Модель таблиці даних Documentation

З реалізацією правильної структури бази даних ми зможемо вирішити проблеми з пошуком, збереженням інформації. Для нормального документування в базі створена таблиця яка буде виконувати роль сховища для створених однотипних звітів та накладних, за необхідності їх можна буде витягнути з бази та точно побачити які дії та роботи, що були проведені. Це також звільняє персонал від ведення документування. Мапа по якій рухається агент завжди є в його прямому доступі для виконання роботи чи простого переміщення по території, а доступ до таблиці з мапою дає персоналу можливість модифікувати її в режимі реального часу, перекриваючи дороги чи навпаки відкривати нові. База з вантажами дозволить проводити з ними маніпуляції і вести підрахунок перевезених вантажів, вона також буде містити унікальну інформацію про конкретний вантаж для облегшення його ідентифікування.

Таким чином ми отримуємо базу даних яка працює методами EF, вона зберігає той об’єм інформації який потрібен нам для роботи агенту і системи. Потім потрібно налаштувати взаємодію бази з другими програмними додатками на кшталт Excel, для мультизадачності. Бази даних створені на основі Entity

Framework працюють швидко, оскільки в самому Entity Framework реалізована асинхронний доступ до даних, оскілки програма буде працювати у зв’язці з C# за допомогою фреймворків ASP.NET та .NET Framework, ми зможемо реалізувати крос платформну взаємодію з базою даних, а також забезпечити роботу з базою даних за допомогою веб додатків. Великим плюсом являться той факт, що Entity Framework дозволяє взаємодіяти з базою за допомогою сутностей entity, це дозволяє писати код для бази даних швидше.

Entity Framework дозволяє значно скоротити код роботи з базами даних. При цьому він надає великі можливості.

Наприклад, можна використовувати:

зовнішні ключі;

зв'язку один-до-одного, один-ко-многим і багато-до-багатьох;

параметри запиту;

збережені процедури;

При звичайній роботі з базою на пряму розробник повинен перейматись про підключення SQL та параметрів про відправлення транзакцій, параметрів та запитів, на EF все це робиться автоматично, програміст працює на пряму з сутностями і тільки каже EF, що потрібно зберегти зміни [11].

3.5 Панель управління оператору

У розробленій системи модель буде виглядати наступним чином, на інформаційному табло оператора буде розташована карта території з помічними на ній зонами та сіткою маршрутів, легенду карти можна буде відкрити по кліку на кнопку редагування легенди, карту також можна буде редагувати в режимі реального часу натиснувши на редагування мапи. На мапі будуть позначені положення усіх агентів з поміткою в якому режимі знаходиться агент, при наведенні на нього буде показана інформація про те який це конкретно агент, його ідентифікаційний номер, тип, роботу яку він виконує.

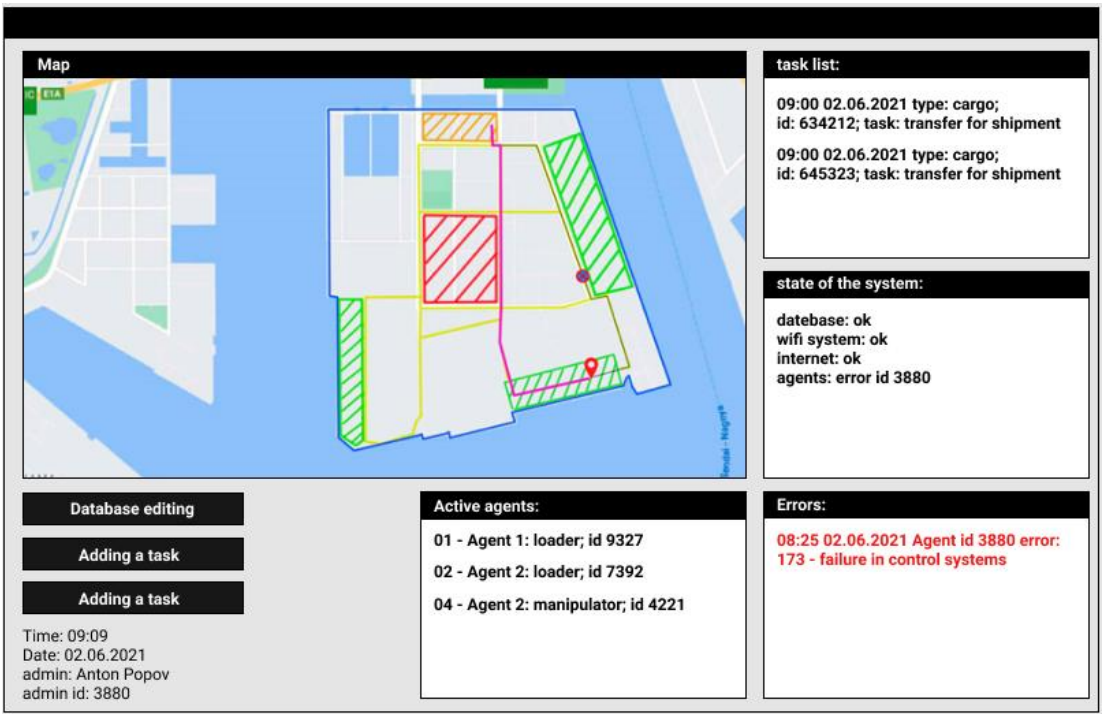

Рисунок 3.12. Модель дизайну системи управління

Зліва від мапи буде розташована таблиця з завданнями які потрібно буде виконати агентам, система автоматично буде знаходити потрібний агент і додавати його в список для виконання. В таблиці стан системи буде записана інформація ку оператор буде використовувати при пошуку неполадок, можна буде вибрати режим тестування конкретного агенту або ж вибрати систему в цілому, буде вказано відносно до якої складової системи виводиться статус, а також позначені в стилі ok, error параметри. В таблицю Error буде виводитись інформація про поточні помилки системи, критичні помилки які призводять до несправносте складових системи будуть виділятись червоним кольором, помилки які не впливають на систему але потребують уваги будуть виводитись чорним кольором.

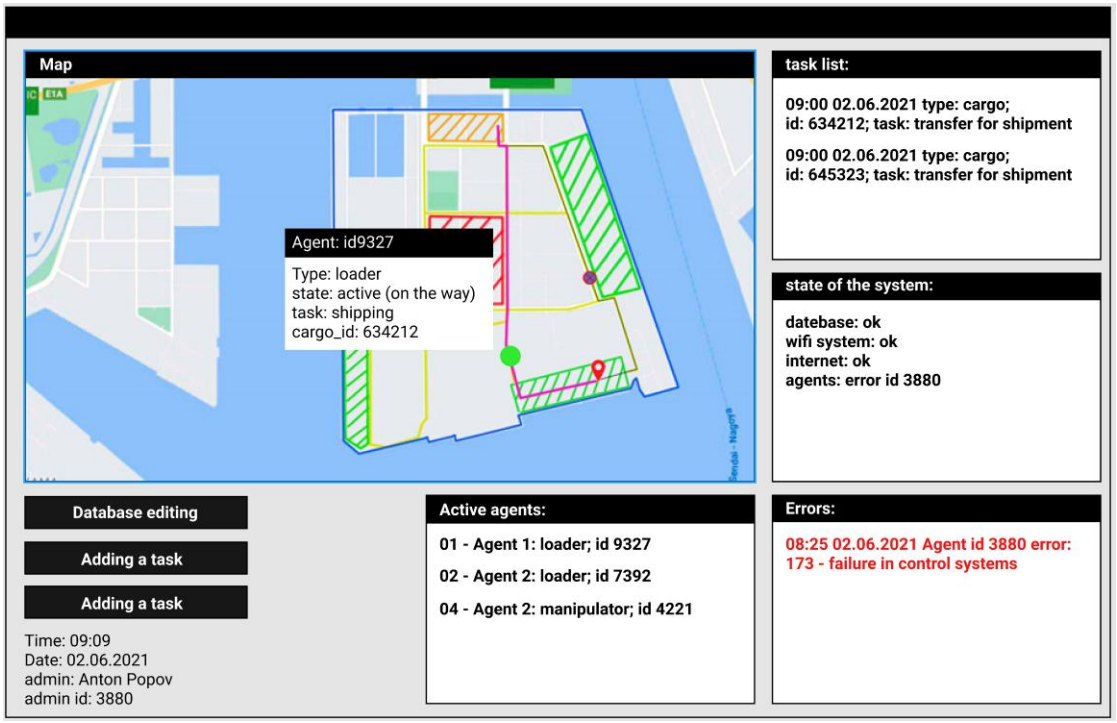

Рисунок 3.13. Вигляд меню при натисканні на конкретний агент

При натисканні на конкретний агент можна побачити інформацію про його стан, його ідентифікатор, чи зайнятий він роботою та побачити конкретний тип завдання який виконує агент. Також можна побачити ідентифікатор вантажу який переміщує агент. Така система потрібна з точки зору нагляду та контролю за системою, навіть повністю автономній системі потрібний нагляд за технічним станом обладнання, за виникаючими похибками оскільки вони можуть призвести до небажаних ситуацій, на кшталт простою системи в пусту у випадку виникнення патової ситуації. Ручна система регулювання з зручним на зрозумілим програмним інтерфейсом допоможе операторові вирішувати певні складні ситуації які будуть виникати під час роботи агенту, також похибки які будуть виникати при роботі будуть зберігатись, документуватись і в подальшому розглядатись з метою покращення робочих функцій системи. Та до повного усунення так званих дитячих проблем, програма по регулюванню роботи агентів необхідна як з огляду на безпечність так і на продуктивність роботи системи в цілому.

Висновок до третього розділу:

В даному розділі ми розглянули яким чином можливо побудувати систему штучного інтелекту та створили схеми процесів які вона буде виконувати, ми побудували модель мережевого з’єднання усіх частин системи і дізналися яким чином можна організувати обмін інформацією в такій мережі, створили модель бази даних та розглянули методи взаємодії з нею. Розглянули методи оптимізації та роботи з нейронною мережею YOLOv4. Створили прототип дизайну для програми оператору.

Обробка зображення на процесорі буде займати купу часу так як процесор аналізує інформацію одно поточно, з використанням відео карти яка аналізує інформацію асинхронно можна виконувати операції в декілька разів швидше. А з використанням CUDA для відео карт NVIDIA, можна добитися результату в сотні, а то і в тисячі разів швидше розпізнавати образи ніж на процесорі.

Для організації нормальної роботи системи знадобиться орендувати хмарний сервіс для розміщення в ньому серверу з СУБД та БД або ж закупити таке обладнання і встановити його в себе на території офісу. Сервер повинен мати швидкі та ємнісні диски для збереження інформації, які до того ж будуть віддзеркалюватись, що хоч і збільшить вагу файлів як мінімум у двоє, зате збільшить безпечність в декілька раз, також можна використовувати резервний сервіс з самою важливою інформацією. Також потрібна установка великої кількості комп’ютерів для ведення контролю над системою.

Зв'язок з сервером агент буде підтримувати по Wi-Fi, це проста у розгортанні та використанні технологія. Головною проблемою таких мереж є легкість до доступу до них, а значить легкість їх вилому, тому потрібно гарно подумати про захист даних, захист від перехоплення управління, вирішити такі проблеми можуть технології шифрування, контролери небезпечного трафіку, та гарно продумана програмна архітектура системи, в якій чорні ходи були виявлені та закриті, ще на етапі проектування, сам Entity Framework сприяє розробці надійних і захищених систем, оскільки більша частина їх відпадає при розробці.

ВИСНОВКИ

1. Розглянуто аналоги існуючих систем для безпілотного управління транспортними засобами та їх застосування в управлінні транспортними засобами в умовах промислового майданчику. Проведений аналіз систем по їх функціональним характеристикам, виявлені слабкі та сильні місця. На основі цього, була сформована чітка задача дипломного проекту. Це система з розпізнаванням образів на основі нейронної мережі, з подальшим обробітком інформації за допомогою фреймворку .NET мови C#, збереження інформації на сервері, за зв'язками між елементами за допомогою мережі WIFI.

2. Була створена принципова схема роботи системи, розглянута взаємодія між її компонентами за допомогою мережі, також були сотворені і розібрані схеми виконання певної роботи штучним інтелектом, були вибрані технології для реалізації роботи системи та підібране апаратне забезпечення для оптимізації роботи системи, також була створена система орієнтування по території для оптимізації роботи алгоритму системи застосованих алгоритм пошуку найкоротшого шляху.

3. Розібрали принципи роботи нейронної мережі YOLO, проведено дослідження з метою оптимізації роботи цієї нейронної мережі, також було проведено дослідження всіх аспектів її роботи, виявлені позитивні стороно цієї нейронної мережі, підібрані технології з якими вона працює більш ефективно.

4. Спроектована база даних з використанням технології Entity Framework, за допомогою якої можна розробляти бази даних працюючи напряму з сутностями без написання SQL коду, були наведені схеми взаємодії різних частин системи з базу даних, схема самої бази даних, описана її структура та неведений програмний код по всіх таблицях бази даних.

У результаті роботи над дипломним проектом була розроблена концептуальна система управління транспортним засобом в умовах промислового майданчику, алгоритми були виконані таким чином, щоб оптимізувати його роботу.