**Міністерство освіти і науки України**

**Харківський національний університет радіоелектроніки**

Факультет комп’ютерних наук

Кафедра Програмної інженерії

**АТЕСТАЦІЙНА РОБОТА МАГІСТРА**пояснювальна записка

Дослідження методів імітаційного навчання для управління автомобілем

Магістрант гр. ПЗСм-16-1 Шпетний Д.В.

Керівник роботи Турута О.П.

Рецензент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Рецензент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Допускається до захисту  
Зав. кафедри, проф. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дудар З.В.

2018 р.

АРКУШ ЗАВДАННЯ (сделать)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до атестаційної роботи: 54 с., 10 рис., 7 табл., 3 додатки, 27 джерел.

МАШИННЕ НАВЧАННЯ, МАШИННИЙ ЗІР, АВТОНОМНИЙ ТРАНСПОРТ, МОДЕЛЮВАННЯ СЕРЕДОВИЩА.

Об’єктом дослідження є автономний автомобіль, аналіз можливостей сучасних технологій автономного керування транспортним засобом, моделювання і дослідження дорожньо-транспортного середовища у контексті дослідження можливостей автономного керування.

Методи розробки базуються на технологіях C#, Unity, Python.

У результаті роботи здійснена програмна реалізація моделі дорожньо-транспортного середовища та досліджені методи електронної комерції для продажу товарів оптичного асортименту і реалізації послуг у мережі Internet для фірми “Інтелект – оптика»

(СДЕЛАТЬ АНГЛ ПОСЛЕ УТОЧНЕНЕНИЯ)

ACCESSORIES, APACHE, CONTACT LENSES, CORRECTION, ELECTRONIC SHOP, INTERNET SOLUTIONS, MySQL, OPTICAL, PHP, SIGHT, WEB-site.

The object of research is an enterprise, which is engaged in the realization of goods and services through the Internet. The “Intellect-optic» firm was chosen as an example, which deals with the contact sight correction sphere and sells contact lenses, their accessories and maintenance liquids.

The aim – the development of an electronic commerce system for selling optical goods through the Internet

Methods of developing technology based on PHP, MySQL database server and Web-server Apache.

Results – the analysis is preformed and the program realization of electronic commerce system for selling goods and services with the use of the Internet for the “Intellect-optic» firm is completed.

**ЗМІСТ (ШРИИИИФТ)**

Перелік умовних скорочень...........................................................................................................

Вступ................................................................................................................................................

1 Назва розділу................................................................................................................................

1.1 Назва підрозділу.................................................................................................................

1.2 Назва підрозділу................................................................................................................

2 Назва розділу................................................................................................................................

2.1 Назва підрозділу.................................................................................................................

2.2 Назва підрозділу.................................................................................................................

2.3 Назва підрозділу.................................................................................................................

2.4 Назва підрозділу.................................................................................................................

3 Назва розділу ................................................................................................................................

3.1 Назва підрозділу.................................................................................................................

3.2 Назва підрозділу.................................................................................................................

4 Назва розділу ................................................................................................................................

4.1 Назва підрозділу.................................................................................................................

4.2 Назва підрозділу.................................................................................................................

Висновки...........................................................................................................................................

Перелік джерел посилання………..................................................................................................

Додаток А Програмний код, схема, алгоритм тощо...…........……….………………......……..

Додаток Б Слайди презентації...............………………….......…………………………………..

Додаток В Апробація результатів роботи...........................………….…………………...……..

Додаток Г Електронні матеріали (CD)....................................................................……….

# ВСТУП

Людство завжди прагне до автоматизації повсякденних завдань. Безсумнівно рутина робота яка може не робитися за рахунок людського часу з часом буде автоматизована тим чи іншим способом. Керування автомобілем є однією з таких задач. Перший інтерес управління засобом транспорту згодом змінюється на ніяковість у зв’язку із монотонністю процесу. Саме тому можно побачити аудіосистеми у сучасних автомобілях – активний процес керування стає напівпасивним на рівні рефлексів. Проте користування індивідуальними засобами транспорту для багатьох є необхідністю. Розробники автомобілів ставлять за одну з найважливіших цілей підвищення комфорту та безпеки керування. Для цього постійно ведуться роботи над покращенням систем контролю та створення автоматичних допоміжних систем для водія (адаптивне керування, режими круїзної подорожі та інше). Такі системи надають конкурентноспроможність у пересиченому ринку автомобілей.

Наступним етапом автоматизації стає усунення водія як основного центру виконування механічних дій для керування автомобілем і перетворення його ролі на контролюючу та спрямовуючу – задання маршруту, обрання типу пересування (пріоритет швидкості, комфорту, безпеки тощо) та вибір інших параметрів вищого рівня абстракції. Окрім полегшення безпосередньо процесу водіння також можливо використовувати технологічні досягнення для підвищення безпеки пересування. Наприклад, керування автомобілем у критичних ситуаціях, де швидкість реакції людини нижча за рівень обчислювальних можливостей комп’ютеру; або пересування за умов поганої видимості, де сукупніть датчиків надасть перевагу над людським зором у кількості та якості інформації.

Над проектом автономного керування працюють усі визначні автомобільні компанії (наприклад, Toyota, Ford, Volkswagen, Tesla Motors) та гіганти розробки програмного забезпечення (Google, Samsung, Nvidia та інші). Однак результати їх роботи не можуть бути використаними у серійних автомобілях саме зараз у зв’язку з юридичними складностями та неповноцінністю систем. Але безумовно даний напрям роботи є одним з провідних для багатьох дослідницьких центрів оскільки надає можливість не тільки отримувати значне фінансування, а й знаходитися на межі вивченого у суміжних наукових сферах.

Основні проблеми у дослідженні автономного пересування у мережах міста є безпосередньо неможливість навчати і тестувати систему фізично у самому місті. Підготовка та операція навіть над єдиним екземпляром автомобілю потребує значних кошт та кваліфікацій у суміжних областях діяльності, що є неможливим для невеликих груп дослідників. Один екземпляр автомобілю надає недостатньо даних для збору та обробки необхідних для покриття усіх крайових сценаріїв. Це стає очевидним навіть при типово алгоритмічному підході до задачі, не кажучи про більш вимогливу до кількості та якості інформації для аналізу концепцію машинного навчання.

Тематика автономного автомобілю поєднує такі сфери науки як машинний зір, машинне навчання, робототехніку та інше, що робить складними комплексні дослідження у даному питанні. Але така складова наукових галузей дослідження також є перспективною з точки зору актуальності новітніх концепцій та поточних досліджень. Тобто існує наукова база разом з економічними інтересами великих компаній, що дає впевненість у подальшому розвитку даного напрямку роботи.

На кафедрі програмної інженерії університету постійно ведуться дослідження у галузях машинного зору, штучного інтелекту та моделювання складних мультиагентних систем, що дає достатню теоретичну базу для виконання даного дослідження.

Метою роботи є дослідження існуючих систем автономного пересування, транспорту та прийняття рішень задля покращення існуючих способів аналізу дорожньо-транспортної ситуації у системах автономного автомобілю завдяки використання додаткових груп сенсорів та новаторських підходів у аналізі сукупності даних. Додатковою метою є дослідження оптимального середовища та технологій для подальшої роботи над основним дослідженням.

Інтеграція систем автоматичного керування є небезпечною у першу чергу завдяки тому що неможливо передбачити та опрацювати усі сценарії поведінки звичним алгоритмічним шляхом, тому є раціональним використовувати деякі практики машинного навчання та впровадження базових концепцій штучного інтелекту, які сформують модель поведінки, що емулює дії професійного водія.

Об’єктом дослідження є автономний автомобіль та створена програмна симуляція обраних транспортних ситуацій. У ході дослідження були використані емпіричні, експериментальні методи дослідження у сукупності з абстрактно-формальним моделюванням середовища. Отримані результати удосконалюють роботу аналогічних систем контролю та аналізу середовища згідно обраним формальним критеріям. Дані наукові інновації можуть бути використаними для поліпшення існуючих систем аналізу та прийняття рішень у автоматизації керування, або інтегровані у автомобільні допоміжні підсистеми контролю та аналізу руху.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

## 1.1 Аналіз предметної області розробки

Автономний автомобіль складається з безпосередньо транспортного засобу, який виконує роль основної платформи, комплексу сенсорів різного типу та програмної системи, що виконує роль агрегатору даних та приймає рішення на базі наявних даних [1]. Приклад такого транспортного комплексу наведений у схематичному вигляді на рисунку 1.1.



Рисунок 1.1 – Схематичний приклад автономного автомобілю

Традиційним підходом до вибору сенсорів є наявність камер та візуальних сенсорів різного формату, що надають недостовірні дані по окремості, але їх сукупність дозволяє відтворити більш дійсний склад речей. Розглянемо наявні сенсори та їх можливості у контексті аналізу дорожньо-транспортної ситуації.

Предметна область розробки знаходиться на перетині галузей робототехніки, прийняття рішення, комп’ютерного зору, машинного навчання. У даному дослідженні зроблений акцент на програмних системах, тому деякі деталі з інших галузей подані у спрощеному вигляді або не розглядаються взагалі.

Типовий склад сенсорів у автономному транспортному засобі [1] поданий на рисунку 1.2 у схематичному форматі.

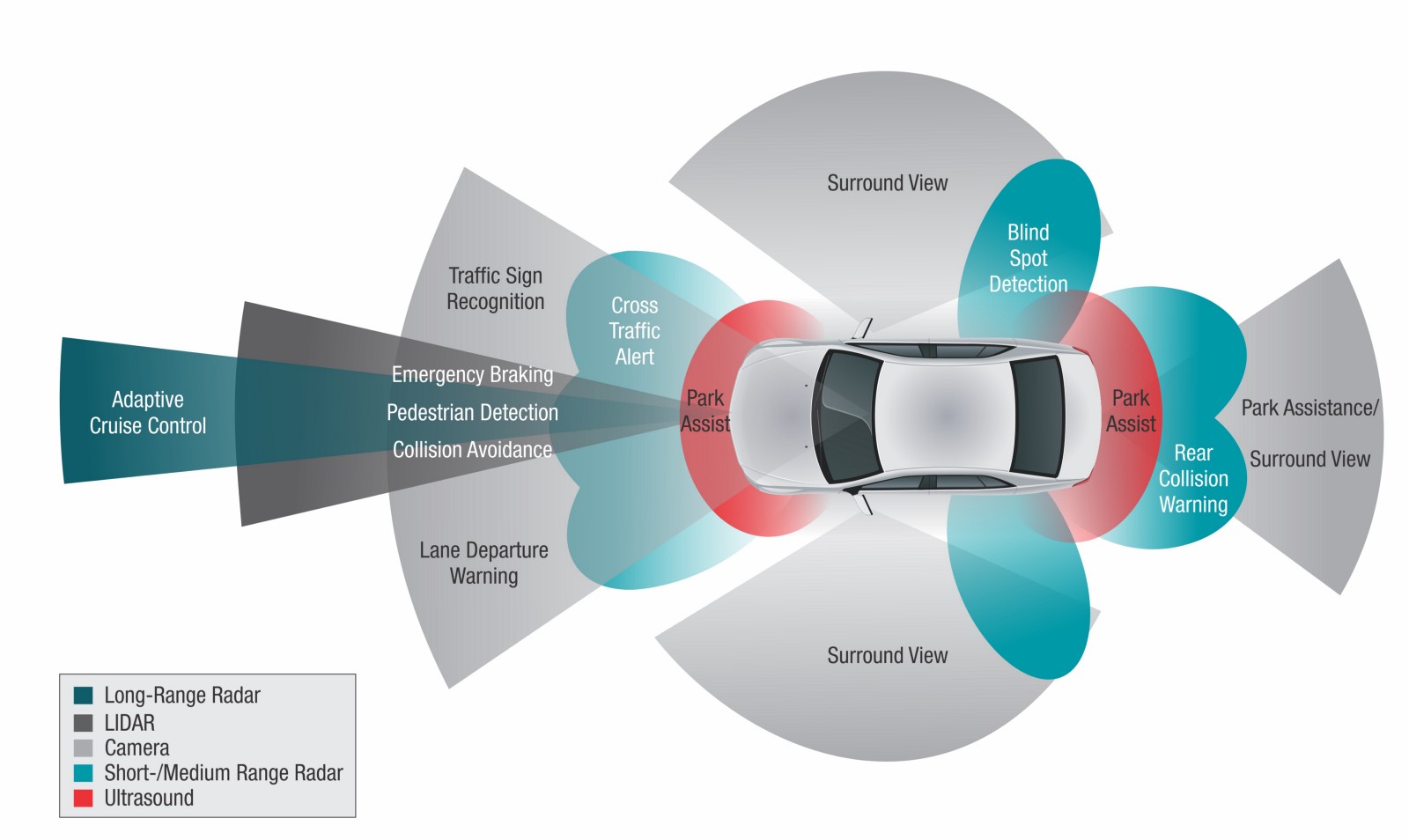


Рисунок 1.2 – Детектори автономного автомобіля

Загалом, можна виділити основні класи проблем, що вирішує кожен с датчиків:

* критичні ситуації – LIDAR, камера, допоміжні радари короткого радіусу дії. До таких ситуацій переважно входять виявлення аномальної поведінки пішоходів або інших динамічних агентів на дорозі таких як автомобілів, тварин, падаючих гілок чи каменів.
* допоміжні інформаційні дані – камери, радари різного радіусу дії, ультразвукові радари. До цього класу можна віднести аналіз типових сліпих зон водія, допомога у аналізі дорожньої ситуації (знаки, світлофори, дорожня розмітка), допомога у парковці.
* системи покращення життя водія – камера, радар дальнього радіусу дії. Переважно це такі системи як круїз контроль, моделювання навколишнього середовища.

LIDAR (Light Identification, Detection and Ranging) – це технологія отримання та обробки інформації про віддалені об'єкти за допомогою активних оптичних систем, що використовують явища відбиття світла і його розсіювання в прозорих і напівпрозорих середовищах. Скануючі лідари в системах машинного зору формують двовимірну або тривимірну картину навколишнього простору [2].

Ключовими з точки зору проектування автономного автомобілю є питання використання даних з сенсорів у реальному-часі, що накладає певні обмеження до складності та оптимізації обчислень. Одними з систем такого роду є комп’ютерні ігри, які можуть складати базу оточення для дослідницьких робот з візуальним зображенням [3].

Окрім безпосередньо потоку інформації також необхідно обробляти його для прийняття рішень у керуванні транспортним засобом, чим займається підсистема аналізу та прийняття рішень.

На сьогодні існує дві основні парадигми для автономних систем водіння на основі комп'ютерного зору: опосередкований підхід до сприйняття (mediated perception approach) щоб розібрати всю дорожню сцену, і підхід рефлекторного аналізу поведінки для прийняти рішення про водіння (behavior reflex approach) яка складає пряму залежність між зображенням та безпосередньою дією завдяки регрессору.

Mediated perception approach складається з багатьох підкомпонентів для розпізнавання об’єктів що відносяться до водіння таких як дорожня розмітка, знаки, лінії руху, світлофори, автомобілі, пішоходи тощо. Результати аналізу складаються у загальну картину безпосереднього оточення автомобілю. Для контролю транспортного засобу система штучного інтелекту бере до уваги всю інформацію перед прийняттям рішення. Оскільки тільки невелика кількість об’єктів має значення у виборі рішення [4], то рівень повного аналізу оточення може додати непотрібний шар складності до і так важкої задачі. На відміну від багатьох задач робототехніки автономний автомобіль контролює лише два параметри – швидкість та напрямок. Остаточний простір рішень має невеликий ступінь свободи на порівняння зі складністю розмірності моделі оточуючого світу. Повнота інформації також надає необхідність використовувати велику кількість сенсорів, що в свою чергу підвищує загальну ціну системи.

Behavior reflex approach складається з прямої залежності між сенсорними вхідними даними та безпосередньо дією автомобіля. Ідея полягає у використанні нейронної мережі для створення прямого мапінгу від зображення до дії. Для навчання використовується реальний водій. Система зберігає зображення та дії водія під час конкретного зображення. Така концепція елегантна у своїй простоті, але вона може зазнавати нижче приведених складностей під час пересування у реальному трафіку та під час виконання складних водійських маневрів. По-перше, водії приймають різні рішення під час однакових ситуацій, що може спричинити розбіжності під час навчання регрессора. Наприклад, якщо автомобіль знаходиться прямо на шляху, то один водій може продовжувати рухатися за машиною, інший – обігнати зліва або справа. Коли всі сценарії присутні у даних, то модель зазнає складностей у прийнятті рішення хоча сенсорні початкові дані однакові. По-друге, система прийняття рішень на такій базі є занадто низькорівневою. Прямий мапінг не може розібратися з повною картиною ситуації. Для прикладу, з точки зору такої моделі обгін автомобілю та пересування назад на полосу є низкою дуже низькорівневих рішень як поворот руля на певний час, а потім поворот у інший бік на певний час. Система не може правильно реагувати у критичних ситуаціях та не може відслідковувати реальний стан речей, які підходять для водіння – необхідний рівень абстракції – це маневр обгону або пересування за автомобілем, а не ступінь повороту руля.

У даній роботі розглядається така модель, що поєднує у собі ці дві концепції. Ця парадигма займає місце між першими двома. Ключовою ідеєю концепції є існування певних станів та індикаторів доступності (affordance indicator) у контексті дорожньої ситуації [5]. Індикатори можуть обиратися в залежності від обраної задачі. Наприклад, дистанція до інших автомобілей, дистанція до дорожньої розмітки тощо.

Важливим компонентом у автономному автомобілі є підхід до штучного інтелекту, який приймає рішення на основі даних. Концептуально можна поділити системи штучного інтелекту на ті що навчаються з учителем (supervised learning) та на самонавчаємі (reinforcement learning).

Традиційним підходом для систем автономного транспорту є використання даних отриманих під контролем людського водія. Це надає змогу отримувати лише коректні дані та спростити кількість необхідного навчання у силу оптимальності контролю автомобіля досвідченим водієм. Недоліками концепції є неможливість моделювання екстремальних ситуацій та можливі протиріччя на великих даних, оскільки кожен водій унікально реагує на дорожню ситуацію. Також процес збору даних накладає потенційні обмеження, оскільки людина вирішує що саме є релевантним у процесі збору вхідних даних.

Навчання з підкріпленням на відміну від першого підходу не потребує людського водія та початкових даних. Основою концепції є встановлення цілей для досягнення та критерії оптимальності їх виконання. Наприклад, для автомобіля це може бути подорож із точки А в точку Б з мінімальною кількістю збитих пішоходів, або з найменшим рівнем витрати пального. Часто використовують комбінацію критеріїв для оптимізації сценаріїв поведінки. Недоліками такого підходу є значно більший час на навчання, складність системи, необхідність проведення ретельного аналізу критеріїв оптимальності.

Згідно з останніми дослідженнями саме навчання з підкріпленням дає найбільш оптимальні результати у задачах з неможливістю передбачити всі сценарії події та з необхідністю реагувати у задане вікно часу [7]. Наприклад, одним з важливих досягнень у цей сфері є перемога AlphaGo над людиною у старовинній грі го. Її особливість саме у непередбачуваності ситуації та у неможливості прорахувати всі варіанти розвитку гри. Впровадження такої системи штучного інтелекту потенційно надає можливість виправити недоліки зазначені у попередньому пункті. Приклад структури системи навчання з підкріпленням наведений на рисунку 1.3 у схематичному форматі.

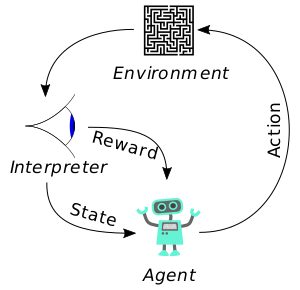


Рисунок 1.3 – Модель reinforcement learning

У області комп’ютерного зору дослідники вивчали кожну підзадачу окремо [8]. Розпізнавання машин та окремих полос руху є основою моделі автономного автомобілю. Типові алгоритми на виході видають обмежувальні геометричні кордони (bounding boxes) машин та лінії (spline) розмежувальних полос.

Розглянемо методи керуючих впливів – тобто засоби контролю автомобілю. На даний момент за допомогою емуляції дій водія доступні педалі керування (педаль прискорення та педаль гальмування), кермо. Для спрощення розглянемо модель автомобілю з автоматичною коробкою передач, де інші системи займаються низькорівневою частиною перемикання та загально контролю над даною частиною автомобілю. Приклад реальної системи емуляції керування наведений на рисунку 1.4 нижче.



Рисунок 1.4 – Система емуляції керування

Дана роботизована система розроблена компанією Ford для повної імітації можливих дій водія-людини. Недоліком такої системи є складність виготовлення та встановлення. Також при використанні моделей такого характеру відпадає можливість застосовувати усі доступні опції налаштування автомобілю. Наприклад, регулювання жорсткості підвіски, прогресивний контроль двигуна і паливної системи та інше.

Тобто для області дослідження автономного руху необхідно комплексно підходити до проблему з урахуванням наведених факторів ризику та недоліків попередніх систем і досліджень. Одним з основних недоліків у даній сфері діяльності безумовно є фінансова складова та неможливість проводити експерименти у реальному світі. Автомобілі коштують дорого, обладнувати навіть один такий усіма необхідними датчиками є неможливим для невеликої дослідницької організації.

В результаті аналізу предметної області було вирішено провести дослідження концептуальних підходів до безпосередньо реалізації автономного автомобілю та систем його навчання на основі комп’ютерної моделі.

## 1.2 Методи аналізу сенсорних даних

Розглянуті вище варіанти датчиків надають різні сукупності даних, але більшість із них є подібними до візуальної інформації або можна розглядати як такий тип. Аналізом візуальних даних займається підрозділ науки про комп’ютерний зір.

Комп'ютерний зір — це загальний набір методів, що дозволяють комп'ютерам сприймати та аналізувати візуальну інформацію Машинний зір є підрозділом інженерії, пов'язаним з обчислювальною технікою, оптикою, машинобудування та промисловою автоматизацією. У даній роботі акцент робиться на програмних методах та технологіях дослідження, а не на інженерних аспектах конкретних пристроїв, тому у тексті будуть деякі спрощення та ідеалізації, які відрізняються від роботи з конкретним комплексом датчиків машинного зору.

Існують деякі типові задачі [17] з аналізу візуальних даних у парадигмі комп’ютерного зору:

а) розпізнавання - один чи декілька попередньо заданих чи вивчених об'єктів або класів об'єктів можуть бути розпізнані, зазвичай разом з їх двовимірним положенням на зображенні чи тривимірним положенням в сцені;

б) ідентифікація - розпізнається індивідуальний екземпляр об'єкта. Наприклад, ідентифікація визначеного людського обличчя, відбитка пальців, автомобіля, тощо;

в) виявлення – візуальні дані перевіряються на наявність визначеної умови. Наприклад, виявлення можливих неправильних клітин чи тканин в медичних зображеннях. Виявлення, що базується на відносно простих і швидких обчисленнях, іноді використовується для знаходження невеликих ділянок в зображенні, що аналізується, які потім аналізуються за допомогою заходів, що потребують більше ресурсів, для отримання правильної інтерпретації.

Дані задачі можуть бути більш детально класифіковані у кожному конкретному випадку їх практичного застосування. У даному дослідженні використовуються переважно методи розпізнавання об’єктів, але також застосовуються і концепції ідентифікації та виявлення.

Зазвичай для отримання необхідної інформації за допомогою методик машинного зору необхідно провести наступні етапи роботи:

а) отримання зображення - цифрові зображення отримуються від одного чи декількох датчиків зображення, які окрім різноманітних типів світлочутливих камер включають датчики відстані, радари, ультразвукові камери тощо. Залежно від типу датчика, отримані дані можуть бути звичайним 2D зображенням, 3D зображенням чи послідовністю зображень. Значення пікселів зазвичай відповідають інтенсивності світла в одній чи декількох спектральних смугах (кольорові чи зображення у відтінках сірого), але можуть бути пов'язані з різноманітними фізичними вимірюваннями, такими як глибина, поглинання чи відображення звукових або електромагнітних хвиль, або ядерним магнітним резонансом;

б) попередня обробка - перед тим, як методи комп'ютерного зору можуть бути застосовані до відеоданих з метою вилучення певної частини інформації, необхідно обробити відеодані, щоб вони задовольняли деяким вимогам залежно від метода, що використовується. Приклади попередньої обробки наведені нижче:

1) повторна вибірка з метою, щоб переконатись, що координатна система зображення є правильною;

2) видалення шумів з метою, щоб видалити спотворення, що вносяться датчиком;

3) покращення контрастності для того, щоб потрібна інформація могла бути виявлена;

3) масштабування для кращого розрізнення структур на зображенні;

в) виокремлення - деталі зображення різного рівня складності виділяються з масиву візуальних даних. Типовими прикладами таких деталей є:

1) лінії та межі;

2) локалізовані точки інтересу, такі як кути, краплі чи точки;

3) деталі структури, форми чи руху;

г) детектування та сегментація - на даному етапі обробки приймається рішення про те, які точки чи ділянки зображення є важливими для подальшої обробки. Наприклад, виділення визначеного набору точок інтересу або сегментація деяких ділянок зображення, які містять певну область інтересу. Цей етап відрізняється від виокремлення складністю рівня аналізу;

д) високорівнева обробка – до цього етапу доходить порівняно невеликий об’єм даних, що аналізується наприклад на належність до певного класу об’єктів; розмір або розташування об’єкту у деякому просторі, тощо.

Для даного дослідження використовувались поєднання методів розпізнавання об’єктів. Так, наприклад, ключовою частиною обробки візуальних даних у даному дослідженні стало виявлення контурів або ліній (edge detection) для ідентифікації кордонів об’єктів у цілому та розмежувальних маркувань дороги.

Виявлення контурів — це назва набору математичних методів, спрямованих на виявлення точок цифрового зображення, в яких яскравість зображення змінюється різко, або, формальніше, має розриви. Точки, в яких яскравість зображення змінюється різко, зазвичай утворюють набір відрізків кривих ліній, що називаються контурами. Аналогічна задача знаходження розривів в одномірних сигналах відома як виявлення сходинок, а задача знаходження розривів сигналу в часі відома як виявлення змін. Виявлення контурів є основним інструментом в обробці зображень, машинному зорі та комп'ютерному зорі, особливо в областях виявлення ознак та виділяння ознак. В ідеальному випадку, результатом застосування до зображення детектора контурів може бути набір з'єднаних кривих, що позначають межі об'єктів, межі забарвлення поверхонь, а також усі криві, що відповідають розривам в орієнтації поверхонь. Таким чином, застосування алгоритму виявлення контурів до зображення може значно зменшувати кількість даних, що підлягають обробці, відфільтровуючи інформацію, яка може розглядатися як менш значуща, але зберігаючи важливі структурні властивості зображення. Якщо крок виявлення контурів є успішним, то подальшу задачу інтерпретування інформаційного вмісту первісного зображення може бути істотно спрощено. Проте не завжди можливо отримувати такі ідеальні контури в картинах реального світу середньої складності.

Контури, виділені з нетривіальних зображень, часто пошкоджено фрагментацією, що означає, що криві контурів не є з'єднаними, відсутні відрізки контурів, а також є хибні контури, що не відповідають досліджуваному явищу в зображенні — ускладнюючи таким чином подальшу задачу інтерпретування даних зображення.

Властивості контурів різняться від типу зображення. Так, наприклад, контури, виділені з двовимірного зображення тривимірної сцени, може бути класифіковано або як залежні від точки огляду (viewpoint), або як незалежні від неї. Незалежний від точки огляду контур зазвичай відображає властивості, притаманні тривимірним об'єктам, такі як забарвлення поверхні та її форма. Залежний від точки огляду контур може змінюватися при зміні точки огляду, і зазвичай відображає геометрію сцени, таку як затуляння одного об'єкту іншим. Типовий контур може бути, наприклад, межею між блоками червоного та жовтого кольорів. З іншого боку, лінія (яку може бути виділено детектором хребтів) може бути невеликою кількістю пікселів іншого кольору на в іншому незмінному тлі. Тому зазвичай для лінії може бути по одному контуру з кожного з її боків.

Ідеалізована модель контурів і ліній складається і розглядається як сукупність з виявлення ідеальних кроків. Проте у реальному світі практично неможливо отримати таку модель оскільки отримані зображення не є ідеальними кроками, а страждають від наступних дефектів: фокусне розмиття, що викликане кінцевою глибиною поля (finite depth-of-field) та кінцевою функцією розподілення точок (point spread function); напівтіні, створені джерелами світла не нульового радіусу; затемнення гладкого об’єкту.

Операція виділення контурів проводилася за допомогою алгоритму детекції границь Кенні (Canny edge detection). Даний алгоритм відноситься до складу баготопрохідних алгоритмів, тому накладає деяку обчислювальну складність, що може стати критичним у системах реального часу. Алгоритм задовольняє таким критеріям як гарний процент виявлення (більш високий рівень відношення сигналу і шуму), гарна локалізація (вірне виявлення положення границі), єдиний консистентний відгук на одну границю. Для задовільнення даних критеріїв алгоритм використовує методології варіаційного числення – техніки, яка знаходить функцію, що оптимізує заданий функціонал. Оптимальна функція у декеторі границь Кенні описується як сума чотирьох експоненцій, але приблизний розрахунок можна виконати за допомогою першої похідної гаусіана [17]. Даний алгоритм складається з наступних кроків:

а) до похідного зображення застосовується фільтр гаусса для згладжування задля видалення шуму;

б) знаходження градієнтів інтенсивності зображення;

в) застосування не максимального стримування для видалення помилкових границь (edge thinning);

г) застосування подвійного порогу для визначення потенційних границь;

д) визначення границь за допомогою гістерезису (залежність стану системи від попередніх значень деяких характеристик системи);

е) фіналізація за допомогою відкидання слабких границь, що не поєднані до визначених сильних границь.

Фільтр Гауса необхідно застосовувати оскільки результати визначення границь сильно залежать від шуму на оригінальному зображенні. Тому необхідно провести фільтрацію такого шуму для видалення помилкових виявлень границь. Нижче наведена формула фільтру Гауса для розміру

|  |  |
| --- | --- |
|  | (1) |

Зазвичай використовується розмір оскільки він задовольняє більшості потреб у попередній фільтрації для даного методу порівняно зі складністю обчислення [17].

Границя у зображенні може знаходитися у різних напрямах, тому даний алгоритм використовує чотири фільтри для виявлення горизонтальних, вертикальних та діагональних границь у попередньо відфільтрованому зображенні. Існує декілька методів виявлення безпосередньо контурів – наприклад, оператор Собеля, оператор Прюітта, оператор Робертса та інші. У даній роботі використовувався оператор Собеля як найбільш оптимальний для даного класу задач [18]. Розглянемо детальніше принцип його роботи. Це дискретний диференціальний оператор, що обчислює наближене значення градієнта чи норми градієнта для яскравості зображення. Оператор Собеля (Собеля-Фельдмана) базується на згортці зображення невеликими сепарабельними цілочисельними фільтрами в вертикальному та горизонтальному напрямках. Хоча, апроксимація градієнта досить груба, особливо на високочастотних ділянках зображення. Оператор використовує ядра 3×3, з якими згортає зображення для обчислення наближених значень часткових похідних по горизонталі та по вертикалі. Якщо вихідне зображення, а та — два зображення, де кожна точка містить часткові похідні по та по відповідно. Вони обчислюються наступним чином:

|  |  |
| --- | --- |
|  | (2) |

Для зображення , та наступним чином для зображення :

|  |  |
| --- | --- |
|  | (3) |

Координата зростає «направо», а — «вниз». Для кожної точки зображення наближене значення градієнта обчислюється через наближенні значення часткових похідних:

|  |  |
| --- | --- |
|  | (4) |

Напрямок градієнта розраховується наступним чисном:

|  |  |
| --- | --- |
|  | (5) |

Де, наприклад, кут Θ рівний нулю для вертикальної границі, в якої темна сторона зліва. Значення функції існує тільки на регулярній сітці, тому, строго кажучи, не можна знаходити похідні, але припустивши неперервність функції можна застосувати скінченні різниці, а саме оператор Собеля для апроксимації часткових похідних. Тому оскільки функція інтенсивності цифрового зображення відома лише в окремих точках, похідні від цієї функції не можуть бути визначені, якщо ми не вважатимемо, що існує основна неперервна функція інтенсивності, яка була відібрана у точках зображення. З деякими додатковими припущеннями, похідна від функції безперервної інтенсивності може бути розрахована як функція на інтенсивності, що відбирається, тобто цифрове зображення. Виявляється, що похідними в будь-якому конкретному пункті є функції значень інтенсивності практично у всіх точках зображення. Проте апроксимації цих похідних функцій можна визначити в менших або більших ступенях точності. Оператор Собеля-Фельдмана являє собою досить неточне наближення градієнта зображення, але все ще є достатньою якістю для практичного використання в багатьох програмах. Приклад дії оператору можна побачити на рисунку 1.5 нижче.

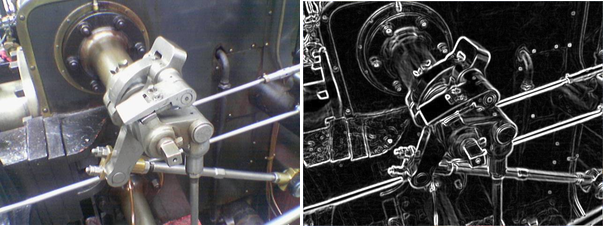


Рисунок 1.5 – Оригінальне зображення та застосування оператора Собеля

Точніше, оператор використовує значення інтенсивності лише в області 3 × 3 навколо кожної точки зображення, щоб наблизити відповідний градієнт зображення, і використовує цілі значення лише для коефіцієнтів, які визначають інтенсивність зображення для отримання градієнтного наближення. Як наслідок його визначення, оператор Собеля може бути реалізований простими способами як на апаратному рівні, так і в програмному забезпеченні: для обчислення відповідного результату потрібні лише вісім точок зображення навколо однієї точки, а для обчислення апроксимації векторного градієнта не потрібна арифметика з дробовими числами.

Наступним кроком є потоншення країв (edge thinning). Дана операція полегшує пошук та розпізнавання об’єкту за рахунок гострих меж. У ідеальному випадку після виконання даної операції можна отримати лінії товщиною у один піксель. У алгоритмі Кенні використовується метод не максимальної супресії (non-maximum suppression). Суть даного методу полягає у видаленні усіх значень градієнту (тобто прирівнення їх до нуля) окрім локального максимуму, який ідентифікує найбільшу зміну у інтенсивності значення. Алгоритм для кожного пікселю зображення наступний:

а) порівняння сили (інтенсивності) краю поточного пікселю з силою краю пікселя у позитивному та негативному напрямку градієнту;

б) якщо ця сила поточного пікселю є найбільшою у порівнянні з іншими пікселями у тому ж самому напрямку, то значення буде збережено. У іншому випадку значення відкидається.

У деяких варіаціях алгоритму аналізується не тільки безпосередньо градієнт наведеним лінійним методом, а відбувається його сегментація на достатньо малі шматки у межах яких проводиться порівняння інтенсивності. Для сегментів обчислюється сумарна інтенсивність і відкидаються найменш значимі сегменти аналогічно наведеному вище алгоритму. Такі імплементація може бути більш оптимальним вибором для деяких конкретних випадків [19].

Після застосування методу потоншення ліній залишкові крайові пікселі забезпечують більш точне відображення реальних границь зображення. Тим не менш, залишаються деякі крапкові пікселі, які обумовлені зміною шуму та кольорів. Для того, щоб врахувати ці помилкові відповідності, важливо відфільтровувати пікселі краю зі слабким значенням градієнта і зберегти пікселі краю з високим градієнтним значенням. Це досягається шляхом вибору високих та низьких порогових значень. Якщо градієнтне значення граничного пікселя перевищує значення високого порогу, воно позначається як сильний піксель краю. Якщо значення градієнта краю пікселя менше, ніж значення верхнього порогу і більше, ніж значення нижнього порогового значення, воно позначається як слабкий піксель краю. Якщо значення пікселя краю менше, ніж значення нижчого порогу, воно буде відкинуте. Два порогових значення визначаються емпірично, і їх визначення залежить від вмісту конкретного набору вхідних зображень.

Наступний етап відсіювання пікселів у поточних краях зображення відбувається за допомогою гістерезису. Реальні лінії границь зображення дійсно присутні на вже даному етапі, але є сумніви відносно слабких пікселів, які можуть бути результатом зміни шуму або кольору. Зазвичай слабкі пікселі, що дійсно є частиною меж приєднані до сильних пікселів меж, тоді як похибкові пікселі не пов’язані з ними [19]. Для відстеження таких зв’язків використовується сегментний аналіз, де розглядається слабкий піксель та сукупність його суміжних пікселів (зазвичай беруть 8 найближчих пікселів). Поки у даному сегменті є хоча б один сильний крайовий піксель, то слабкий піксель зберігається.

Канонічна реалізація алгоритму Кенні забезпечує порівняно просту, але точну методологію для проблеми виявлення границі. Проте з більшими вимогами до точності і надійності виявлення наданий алгоритм не зможе впоратися з складними підвидами задач. Основні недоліки канонічної реалізації наведені нижче:

а) фільтр Гауса, який застосовується для першочергового позбавлення від шумів також пом’якшує границі, які є характеристикою зображення з високим пріоритетом. Це підвищує шанс пропустити слабкі границі та збільшує кількість відокремленних границь у кінцевому результаті;

б) для розрахунку амплітуди градієнта, алгоритм розпізнавання краю використовує центр у невеликому сегменті квадрату розміром 2 × 2, щоб обчислити середнє значення кінцевої різниці. Цей метод дуже чутливий до шуму і може легко виявити помилкові границі разом з втратою реальних границь;

в) у традиційному алгоритмі виявлення краю для фільтрації помилкових країв буде два фіксованих глобальних порогових значення. Однак, як тільки зображення стає більш складним, різні місцеві області потребують різних порогових значень для точного знаходження реальних країв. Крім того, глобальні порогові значення визначаються вручну за допомогою експериментів за традиційним методом, що призводить до складності і суб’єктивності розрахунку.

Приклад роботи алгоритму наведений на рисунку 1.6 нижче. Для порівняння і більшої наглядності використане оригінальне зображення з рисунку 1.5.

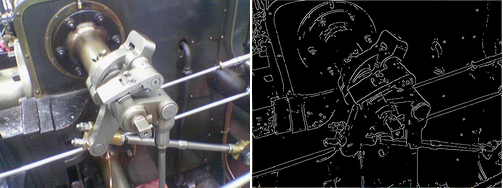


Рисунок 1.6 – Результат роботи алгоритму Canny edge detection

Як можна побачити з поданого зображення, то кількість ліній значно зменшилась після етапів фільтрації. Якість меж також змінилась. Точніше видимі сильні лінії об’єкту, відсіялись несуттєві границі пов’язані зі зміною кольору та текстурою оточення. Проте на даному зображенні все ще присутні лінії та окремі крапки не пов’язані з основною областю інтересу.

Виявлення границь зображення не є кінцевою метою аналізу візуальних даних. Часто необхідно ідентифікувати об’єкт. Розглянемо даний концепт на прикладі ідентифікації ліній дорожньої розмітки. Для цього застосовується метод під назвою перетворення Хафа (Hough transform). Даний метод застосовується у аналізі зображень, комп’ютерному зору та інших суміжних галузях науки. Він відноситься до классу виокремлючих особливості зображення (feature extraction methods), його мета полягає у тому щоб виявити окремі випадки об’єктів у просторі певного класу об’єктів шляхом процедури голосування. Ця процедура голосування виконується у деякому просторі параметрів, з якого об’єкт-кандидат отримують у так званому акумуляторному просторі, що явним чином конструюється для даного алгоритму. Класичне перетворення Хафа застосовується для виявлення ліній на зображенні, але пізніше перетворення Хафа було поширене на виявлення структур довільної форми, найчастіше кругів або еліпсів. Мотивацією для розробки даного метода стало те, що при автоматичному аналізі цифрових зображень найчастіше виникає підпроблема виявлення простих форм, таких як прямі лінії, кола або еліпси. У багатьох випадках детектор границь (edge detection) може бути використаний як етап попередньої обробки для отримання точок зображення або пікселів зображення, які знаходяться на потрібній кривій у просторі зображення. Внаслідок недоліків в даних зображення або детектору краю, однак, на потрібних кривих можуть бути відсутні точки або пікселі, а також просторові відхилення між ідеальною лінією, колом, еліпсом та помилковими точками краю. З цих причин часто не є тривіальним завданням об'єднання витягнутих крайових точок до більш абстрактного відповідного набору рядків, кругів або еліпсів. Мета трансформації Хафа полягає в тому, щоб вирішити цю проблему, дозволяючи здійснювати угруповання краєвих точок в об'єктних кандидатів, виконуючи явну процедуру голосування над набором параметризованих об'єктів зображення.

Розглянемо найпростіший випадок застосування трансформації Хафа – ідентифікація прямих ліній. Як відомо, пряма лінія може бути зображена наступним рівнянням y = mx + b та представлена як точка (b, m) у просторі параметрів. Головна ідея трансформації Хафа полягяє у тому, щоб врахувати характеристики прямої не як рівняння, що побудоване по парі точок зображення, а у термінах параметрів прямої. Тобто m – коефіцієнт нахилу, b – точка перетину з віссю ординат. Проте вертикальні лінії, які є паралельними вісі ординат, постановлюють деякі проблеми. Вони мають безкінечне значення коефіцієнту m. Тому для обчислення запропоновано використовувати нормальну форму Хеса (Hesse normal form). Нижче на формулі шість наведена нормальна форма прямої:

|  |  |
| --- | --- |
|  | (6) |

Де r – дистанція між початком координат і найближчою точкою прямої лінії, а θ – кут між віссю x та лінією, що поєднує початок координат з цією найближчою точкою. Ілюстрація наведена нижче на рисунку 1.7.

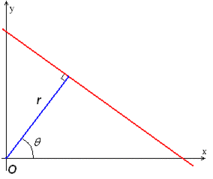


Рисунок 1.7 – Ілюстрація нормальної форми Хеса

Тому можливо встановлювати залежність кожної лінії зображення парі (r, θ). Площину (r, θ) також іноді називають простором Хафа для набору прямих ліній у двовимірному просторі. Таке зображення робить трансформацію Хафа концептуально близькою до перетворення Радона (Radon transofm) у двовимірному просторі. Враховуючи єдину точку площини, сукупність всіх прямих, що проходять через цю точку, відповідає синусоїдальної кривій у площині (r, θ), яка є унікальною для цієї точки. Набір з двох або більше точок, що утворюють пряму лінію, буде створювати синусоїди, які перетинаються на (r, θ) для цієї лінії. Таким чином, проблема виявлення колінеарних точок може бути перетворена в проблему пошуку паралельних кривих [20]. Приклад трансформації Хафа для виявлення ліній наведений на рисунку 1.8 нижче.



Рисунок 1.8 – Виявлення ліній за допомогою перетворення Хафа

На даному зображені лінії, що були обрані попереднім алгоритмом виявлення границь позначені тонше, а фінальні «сильні» лінії позначені товще. Як можна побачити з даного зображення, ця методологія є достатньо непоганою для виявлення дорожньої розмітки та кордонів дороги взагалі.

Також для даної операції визначення полоси руху та лінії дорожньої розмітки взагалі можна використовувати методології, що пов’язані не лише на чистих концепціях аналізу за допомогою технік машинного зору, а застосовувати нейроні мережі глибокого навчання. Такий підхід є більш складним з обчислювальної точки зору, але надає кращі результати в умовах достатньої кількості візуальних даних та тренувальних датасетів [23]. У даній роботі було вирішено не використовувати подальше ускладнення даного сегменту, оскільки термін дослідження та наявні ресурси обмежені.

1.3 Аналіз комп’ютерних моделей дорожньо-транспортної системи

У даний час існує велика кількість досліджень автономних автомобілей та пов’язаних суміжних галузей, але більшість з них не є комплексними системами або є частиною комерційної таємниці, оскільки перспективи досліджень та потрібні економічні інвестиції недоступні невеликим дослідницьким організаціям. Далі детально розглянемо та проаналізуємо системи моделювання оточуючої середи для створення моделі автономного пересування, обрані для дослідження.

CARLA – це симулятор взаємодії машини та оточуючої середи з відкритим кодом. Розроблений спеціально для досліджень автономного автомобіля з урахуванням розробки, навчання, валідації в умовах міського пересування. Екранні форми симулятора наведені на рисунку 1.9 за різних умов симуляції.



Рисунок 1.9 – Симулятор CARLA

Платформа надає деяку кількість візуальних ресурсів, реалізацію частини наведених у розділі аналізу предметної області сенсорів. Але дана платформа призначена для більш масштабних симуляцій тому є з значним ускладненням на порівнянні з необхідними функціями. Також стек технологій розробки накладає певні складності у роботі з симулятором.

GTA V – як не здається дивним, але саме сучасні комп'ютерні ігри є ідеальним середовищем для дослідження деяких систем автономного керування. Приклад фронтальної камери на автомобілі у грі наведений на рисунку 1.10, такий варіант можна застосовувати для тренування візуальної частини регресора.



Рисунок 1.10 – Екранна форма процесу водіння у GTA V

Така система середовища дозволяє вивчати процес керування у форматі чорного ящику, а складність гри знімає необхідність додатково моделювати поведінку середовища (пішоходи, транспорт). Недоліком безумовно є відсутність коду гри та неможливість модифікувати початкові умови за межами гри. Також під питанням є використання гри у академічних цілях довготривалого характеру.

TORCS – симулятор автомобільних гонок з відкритим кодом, що працює під Linux-подібними системами, OS X, AmigaOS 4, Windows. Дана симуляція розроблена за допомогою C++, приклад екраної форми можно побачити на рисунку 1.11. Дана симуляція є не перспективною з огляду на графічну складову.

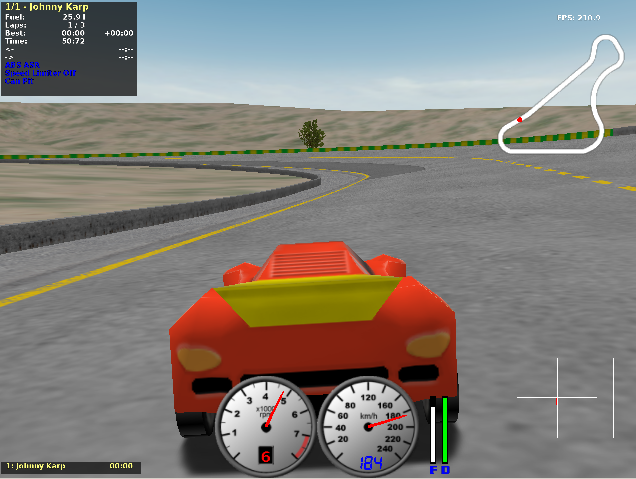


Рисунок 1.11 – Екранна форма процесу водіння у TORCS

Згідно виявлених недоліків існуючих симуляцій було вирішено розглядати розробку власної системи симуляції, яка вдовольняла б потребам саме нашої дослідницької кафедри. Дана реалізація повинна бути масштабовною, мати різні налаштування рівню графічного відображення, повинна мати можливість додавання різних сценаріїв оточення. Прототип системи можно побачити на рисунку 1.12.



Рисунок 1.12 – Екранна форма власного симулятора водіння

Власна реалізація дозволяє мати широкий контроль над необхідними потребами, швидко робити модифікаціїї системи необхідні саме для конкретного дослідження. За бажанням можливо доробляти систему у рамках вивчення курсів моделювання та вивчення програмування, оскільки розробка базується на популярних технологіях, які цікаві студентам.

# 2 ТЕОРЕТИЧНІ ДОСЛІДЖЕННЯ

## 2.1 Концептуальний аналіз проблеми

Визначено, що водії керують автомобілем засновуючись на наступних елементах оточення: дорожня розмітка, спеціальні сигнали (світлофори, знаки та інше), пасивні та активні агенти руху (машини, недоліки дорожнього полотна, паркани, тощо).

Розглянемо можливість автономного руху автомобілю спираючись на інформацію з елементів оточення. Гіпотеза складається у тому, що можливо виділити деякі елементи навіть з однієї камери і за допомогою системи штучного інтелекту приймати рішення про рух спираючись на поточний стан цих елементів.

Розглянемо ідеальні умови пересуваня – рівна дорога, відсутність інших агентів руху, сонячна погода, далека видимість, чіітка дорожня розмітка. Припустимо, що напрямок руху відомий. Тоді можна притримуватися його за допомогою слідкуванння за лініями дорожньої розмітки. Полоси чітко надають водію знання про обмеження руху, тобто вибравши полосу необхідно притримуватися її. Якщо автомобіль лівіше полоси, то необхідно повернути направо і навпаки. Напрями ліній можна дізнатися завдяки дотичній до ній. Скористаємося отриманними знаннями з ідентифікації об’ектів на зображенні, розглянуті у другому підрозділі першого розділу. Необхідно виділити дві обмежувальні лінії з поточних даних сенсорів автомобілю. Для цього знайдемо дві найбільш сильні лінії у області інтересу автомобілю. Запропоновані лінії дорожньої розмітки завжди знаходяться нижче автомобіля, що робить можливим емпірічним чином попередньо визначити регіон інтересу для пошуків в залежності від конфігурації сенсорів. Такий підхід значно спростить процес ідентифікації та обчислення за рахунок попереднього планування налаштування системи сенсорів автомобілю.

Нижче на рисунку 2.1 наведена екрана форма симулятора водіння разом з аналізом ліній дорожньої розмітки.



Рисунок 2.1 – Ідентифікація обмежувальних ліній

Оскільки експериментально доведена можливість роботи даного алгоритму, то можна починати збирати дані для аналізу та модулю машинного навчання, який на сформує модель видимих даних та реакції на них. Тож для подальшого дослідження необхідно скласти датасет, що буде складатися з візуальної інформації (екранна форма у обраний момент часу) як вхідного параметру, та з команд для водіння як вихідного параметру. Технічні способи реалізації будуть розглянуті детальніше у наступному розділі. Однією з важливих проблем є розмір гранулярної одиниці даних для навчання [23]. У однаковий розмір файлу даних може вкластися декілька об’єктів великого розміру або багато об’єктів малого розміру. Для даного дослідження важливо розібрати більшу кількість варіантів сценарія події, що можна зробити лише за рахунок детальності даних в умовах обмежених обчислювальних ресурсів.

В умовах ідеального дорожньо-транспортного оточення, що зазначені вище, дії водія для керування автомобілем мінімальні. Для запропонованого концепту навчання в умовах симуляції це означає, що 80% часу автомобіль просто рухається вперед з мінімальними поворотами убік. Це означає, що необхідно сбалансувати зібраний датасет задля більш точної роботи нейронної мережі. Балансування мережі відбувається завдяки відкиданню частини даних, що можуть спричинити занадто значний поштовх до деяких рішень (biased action) тільки завдяки кількості даних за даними подіями.

Для даної роботи вирішено було застосовувати згорткову нейронну мережу (convolutional neural network) з огляду на візуальний тип даних, що використовується у роботі [24]. Даний клас мереж відноситься до глибинних штучних нейронних мереж прямого поширення. Згорткові нейронні мережі використовують різновид багатошарових перцептронів, розроблений так, щоби вимагати використання мінімального обсягу попередньої обробки. Вони відомі також як інваріантні відносно зсуву (shift invariant) або просторово інваріантні штучні нейронні мережі (space invariant artificial neural networks), виходячи з їхньої архітектури спільних ваг та характеристик інваріантності відносно паралельного перенесення [24]. Даний вибір залишає багато простору для налаштування. Можна обрати слої, кількість нодів у кожному слою, функції активації та інше. Для першої ітерації була обрана модель AlexNet, яка є найбільш точною у порівнянні за своїми альтернативами у області розпізнавання зображень [25]. Дана модель складається з восьми шарів, з яких п’ять є шарами згортки. Використовувалася реалізація на базі бібліотеки машинного навчання TensorFlow. Технічні деталі конкретної реалізації та експерименту будуть розглянуті у наступному розділі.

На даному етапі дослідження стало зрозумілим, що необхідно імітувати процес водіння як керування пропорційно-інтегрально-диференціальним пристроем (proportional–integral–derivative controller) – це механізм зворотного зв'язку, який широко використовується в промислових системах управління та інших додатках, що вимагають безперервного модуляційного контролю. ПІД-регулятор постійно обчислює значення помилки *e(t)* як різницю між бажаною заданою величиною (SP) і вимірювальною перемінною технологією (PV) і застосовує корекцію на основі пропорційного, інтегрального та похідного термінів (позначені відповідно P, I і D), які дають контролерові його назву. У практичному плані він автоматично застосовує точне та чутливе виправлення до функції керування. Щоденним прикладом є круїз-контроль на дорожньому транспортному засобі; де зовнішні впливи, такі як градієнти, можуть спричинити зміни швидкості, а водій зможе змінити бажану задану швидкість. Алгоритм PID відновлює фактичну швидкість до потрібної швидкості оптимально, без затримки або перевищення, керуючи потужністю двигуна транспортного засобу. Без даного контролеру стає складнішим завдавати дії корекції руху, що будуть сприятливими у реальному світі. Наприклад, поворот праворуч є плавним рухом, а не дискретною різкою операцією.

Для розрахунку ситуації на дорозі також важливо розуміти, які об’єкти присутні у даних с з сенсорів. Приклад розпізнавання наведений на зображенні 2.2 нижче.



Рисунок 2.2 – Приклад роботи розпізнавання об’єктів

Для вирішення цієї ситуації неоптимально самостійно вирішувати дану проблему, оскільки потребується значний об’єм робот спрямований на аналіз великої кількості даних, що зможе згодом надавати достовірну інформацію про клас об’єкту. Тому було вирішено скористуватися модулем розпізнавання TensorFlow, який усередині складається з нейронної мережі, що натренована на колосальному обсязі даних. Основним зауваженням було те, що для тренування використовувались зображення з реального світу, які будуть відрізнятись від даних з сенсорів отриманих у результаті роботи симулятора. Виявилось, що безумовно є кореляція між рівнем графічної деталізації та точністю виявлення об’єктів. Результат для GTA V на високому рівні графічної деталізації можна побачити на рисунку 2.3 нижче.



Рисунок 2.3 – Розпізнавання об’єктів у GTA V

Результат є більш ніж задовільним маючи на увазі роздільну здатність даного графічного вікна. Зазвичай необхідна більша деталізація для роботи з об’єктами на великої дистанції. Проте тут виявлена навіть людина, що знаходиться на задньому плані. Це підтверджує запропоновану гіпотезу і дозволяє користуватися даним методом для ідентифікації інших об’єктів інтересу.

Для обчислювання запропонованого коефіцієнту безпеки руху необхідно також оцінювати дистанцію до об’єктів. Проте дана оцінка сильно залежить від реальних фізичних габаритів об’єктів. В силу характеру отриманих візуальних даних репрезентація реального світу виглядає як двовимірне зображення, що накладає певні дефекти та необхідність брати до уваги перспективу зображення графічного сенсору. Для початку аналізу ідентифікуємо такі активні дорожньо-транспортні агенти як легкові автомобілі, автобуси, багатогабаритні вантажівки. Єдиний спосіб визначити розмір – це обчислити кількість пікселів, що займає об’єкт. У даному випадку будемо оперувати відносною відстанню до об’єкту та відносним розташуванням об’єкту. Згідно емпіричним спостереженням, колізія може відбутися при швидкому русі об’єкту до поточного агенту і не відбувається навіть якщо об’єкт відносно близько до агенту, але збоку від нього, що трапляється під час паралельної поїздки. Приклад наведений на рисунку 2.4 нижче.



Рисунок 2.4 – Імовірне зіткнення

У даному випадку об’єкт достатньо близько до поточного агенту та знаходиться у регіоні інтересу, який знаходиться переважно у центрі екрану. Дана концепція регіону інтересу виправдовує себе для руху у ситуації наведеній на рисунку 2.5, але безумовно має певні недоліки.

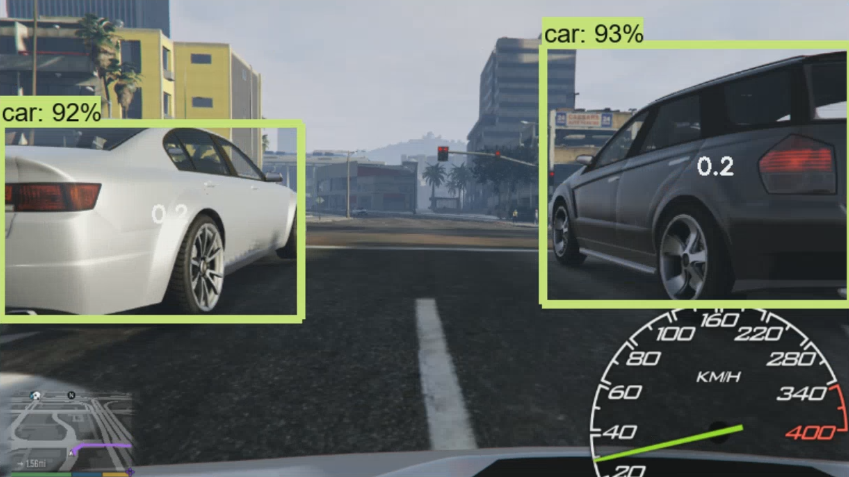


Рисунок 2.5 – Паралельний рух

Загалом можна зазначити, що гіпотези дослідження підтвердилися, що означає теоретичну можливість інтеграції методу оцінки безпеки руху у сучасні автомобілі використовуючи лише достатньо недорогий візуальний сенсор, що паралельно може бути використаний як відеорегістратор. Розглянемо деталі технічної реалізації запропонованих алгоритмів та систем у наступному розділі.

## 2.2 Реалізація

Для дослідження запропонованих гіпотез та впровадження експериментів використовувалась мова програмування Python оскільки у екосистемі присутні інструменти для роботи з бібліотеками OpenCV, TensorFlow та іншими науковими пакетами, що виявилися корисними для використання (такими як, наприклад, Numpy для обробки великої кількості даних).

Python – це інтерпретована об'єктно-орієнтована мова програмування високого рівня з строгою динамічною типізацією. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднання існуючих компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується кілька парадигм програмування, зокрема: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована [11].

OpenCV (Open Source Computer Vision Library), — це бібліотека комп'ютерного зору з відкритим кодом, де сгруповані функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом. Бібліотека надає засоби для обробки і аналізу вмісту зображень, у тому числі розпізнавання об'єктів на фотографіях (наприклад, осіб і фігур людей, тексту тощо), відстежування руху об'єктів, перетворення зображень, застосування методів машинного навчання і виявлення загальних елементів на різних зображеннях. Використання даної бібліотеки дозволяє сконцентруватися на високорівневії частині дослідження і не реалізовувати відомі алгоритми власноруч, що позитивно впливає на швидкість та якість аналізу завданої тематики. У конкретному випадку використовувалась офіційна реалізація (технічно, binding, оскільки внутрішня реалізація бібліотеки створена на C++) OpenCV для мови програмування Python, яка розповсюджується під ліцензією MIT і є безкоштовною для академічного використання.

TensorFlow — відкрита програмна бібліотека для машинного навчання цілій низці задач, розроблена компанією Google для задоволення її потреб у системах, здатних будувати та тренувати нейронні мережі для виявлення та розшифровування образів та кореляцій, аналогічно до навчання й розуміння, які застосовують люди. Її наразі застосовують як для досліджень, так і для розробки продуктів Google, часто замінюючи на його ролі її закритого попередника, DistBelief. TensorFlow було початково розроблено командою Google Brain для внутрішнього використання в Google, поки її не було випущено під відкритою ліцензією Apache. Як і у випадку OpenCV даний фреймворк забезпечує прикладний програмний інтерфейс для мови програмування Python що вписується у запропоновану екосистему розробки. Завдяки відкритій ліцензії, даний фреймворк можна безкоштовно застосовувати у академічних дослідженнях. Використання даного фреймворка дозволяє не реалізовувати відомі алгоритми машинного навчання і використовувати стабільну реалізацію разом з великою базою даних для навчання, що безкоштовно надаються разом з фреймворком. Для конкретного випадку використовувались методи машинного навчання на основі тренувальних датасетів для керування автомобілем та розпізнавання класів об’єктів.

Симуляція була побудована за допомогою ігрового движка Unity та мови програмування C#. Даний вибір технологій обґрунтовується їх популярністю, стабільністю роботи та загальною простотою використання у контексті даної задачі. Згідно проведеному у попередніх розділах аналізу виявлені недоліки у основних конкурентів симуляції дорожньо-транспортного руху. Для вирішення даних проблем було запропоновано реалізувати свій варіант імітації урбаністичного оточення, що просто налаштовувати під кожну конкретну задачу дослідження.

Тобто основною метою архітектури було забезпечити модульну компонентну архітектуру. Загальна архітектура Unity наведена на рисунку 2.6.

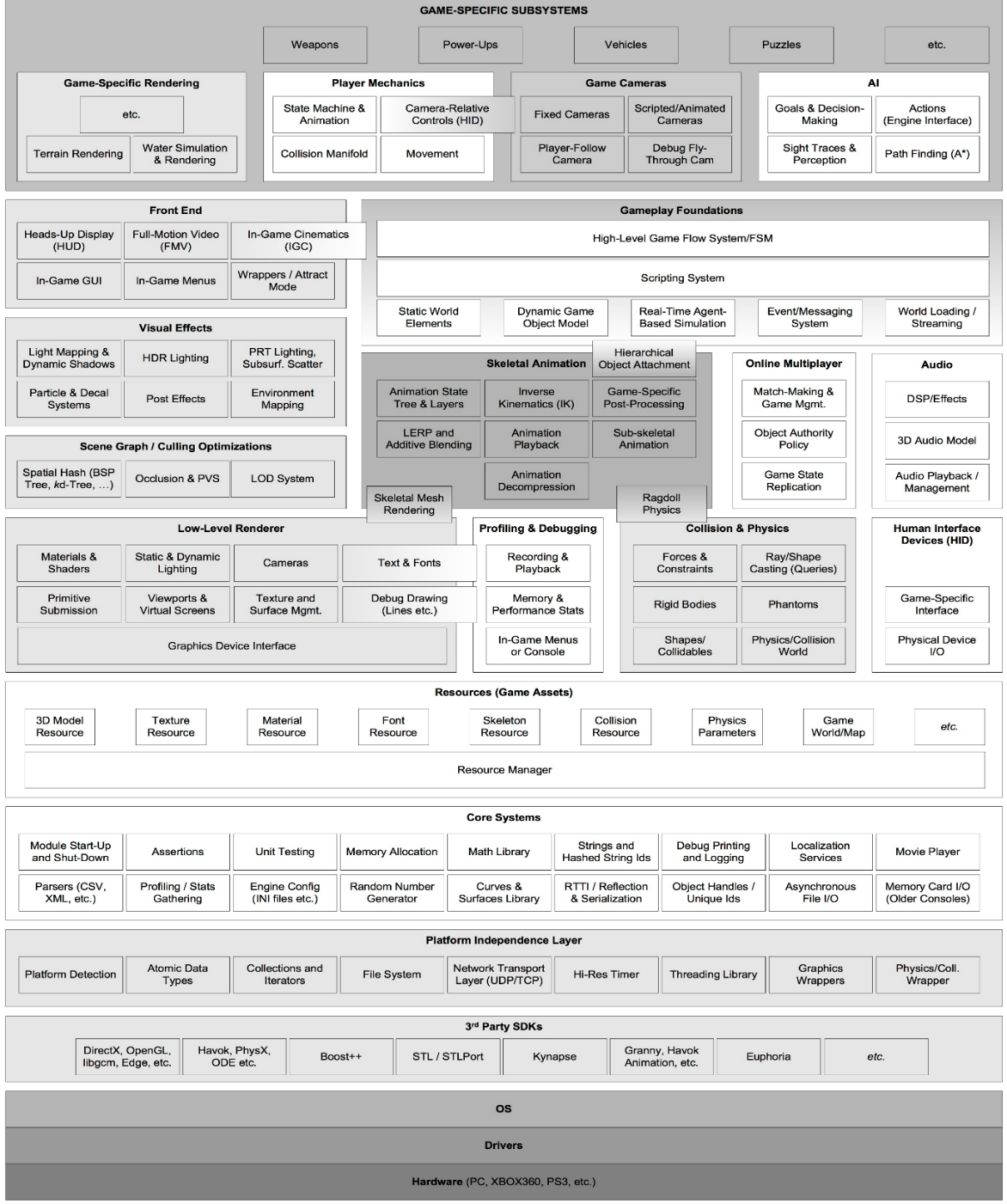


Рисунок 2.6 – Архітектура Unity

Unity – це багатоплатформовий інструмент для розробки тривимірних додатків та ігор для операційних системам Windows, OS X, .Android, iOS, Linux. Також є можливість створювати інтернет-додатки за допомогою спеціального під'єднуваного модуля для браузера Unity, а також за допомогою експериментальної реалізації в межах модуля Adobe Flash Player. Застосування, створені за допомогою Unity, підтримують DirectX, OpenGL, Vulkan [9].

Головна мова програмування у Unity – це C#. C# - об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET. Синтаксис C# близький до С++ і Java. Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. C# розроблялась як мова програмування прикладного рівня для CLR і тому вона залежить, перш за все, від можливостей самої CLR. Це стосується, перш за все, системи типів C#. Присутність або відсутність тих або інших виразних особливостей мови диктується тим, чи може конкретна мовна особливість бути трансльована у відповідні конструкції CLR. Збірка сміття не реалізована в самому C#, а проводиться CLR для програм, написаних на C# точно так, як і це робиться для програм на VB.NET, J# тощо. [10].

Однак однією з особливостей Unity є велика кількість підтримуваних платформ. Тому мова програмування всередині самого ігрового рушія не є повною копією реалізації від Microsoft [10]. Такий підхід необхідний задля коректної роботи на багатьох платформах та для можливості подальшої оптимізації ігрового коду. Подальша оптимізації можлива оскільки мова програмування створюється як інструмент широкого профілю, а у контексті ігрового рушія мова стає провідником логіки, яку реалізує безпосередньо сам рушій.

Важливою частиною у виборі технологій було питання ліцензування та ціни за використання. Даний ігровий рушій є повністю безкоштовним для некомерційних розробок, тобто є безкоштовним і для поточних академічних досліджень. Обрана сукупність інструментів є популярною для розробки ігрових додатків та для програмування у цілому, що збігається з навчальним курсом університету та інтересами студентів. Для даного дослідження це означає, що бажаючі долучитися будуть мати необхідну мінімальну кваліфікацію та зможуть знайти рішення більшості типових проблем пов’язаних саме з технологією, що у свою чергу надасть більше часу на безпосередньо дослідження. Проте у ході розробки прототипу виявилось, що є неоптимальним відмовлятись від запропонованих аналогів на даному етапі роботи в силу складності розробки детального фотореалістичного урбаністичного оточення за умов обмеженого часу та ресурсів розробки. Проте запропонований прототип є повністю працездатним та придатним для подальших допрацювань.

Ключовими задачами експериментальної реалізації запропонованих гіпотез для вирішення стали виявлення візуальних даних з симуляції, передача активуючих дій для транспортного агенту. Виділення візуальних даних відбувалось за допомогою сукупності некросплатформених методів у силу певних вимог кожної операційної системи. Основна платформа для проведення експериментального дослідження – Windows, де застосовувався наступний метод:

**import cv2**

**import numpy as np**

**import win32gui, win32ui, win32con, win32api**

**def grab\_screen(region=None):**

**hwin = win32gui.GetDesktopWindow()**

**if region:**

**left,top,x2,y2 = region**

**width = x2 - left + 1**

**height = y2 - top + 1**

**else:**

**width = win32api.GetSystemMetrics(win32con.SM\_CXVIRTUALSCREEN)**

**height = win32api.GetSystemMetrics(win32con.SM\_CYVIRTUALSCREEN)**

**left = win32api.GetSystemMetrics(win32con.SM\_XVIRTUALSCREEN)**

**top = win32api.GetSystemMetrics(win32con.SM\_YVIRTUALSCREEN)**

**hwindc = win32gui.GetWindowDC(hwin)**

**srcdc = win32ui.CreateDCFromHandle(hwindc)**

**memdc = srcdc.CreateCompatibleDC()**

**bmp = win32ui.CreateBitmap()**

**bmp.CreateCompatibleBitmap(srcdc, width, height)**

**memdc.SelectObject(bmp)**

**memdc.BitBlt((0, 0), (width, height), srcdc, (left, top), win32con.SRCCOPY)**

**signedIntsArray = bmp.GetBitmapBits(True)**

**img = np.fromstring(signedIntsArray, dtype='uint8')**

**img.shape = (height,width,4)**

**srcdc.DeleteDC()**

**memdc.DeleteDC()**

**win32gui.ReleaseDC(hwin, hwindc)**

**win32gui.DeleteObject(bmp.GetHandle())**

**return cv2.cvtColor(img, cv2.COLOR\_BGRA2RGB)**

Даний спосіб виявився більш оптимальним з точки зору продуктивності і оптимальності обчислення у даному конкретному випадку, що дало змогу паралельно запускати симуляцію з ліпшими графічними налаштуваннями. Це в свою чергу дало змогу використовувати розпізнавання об’єктів з більшою точністю в силу попереднього тренування на об’єктах реального світу.

Наступною проблемою виявилось імітування пристроїв вводу/виводу, оскільки деякі з вищеописаних симуляцій працюють напряму з контролерами на низькому рівні. Тому виявилось необхідним реалізувати свій абстрактний модуль такої імітації. Для цього частково були використані функції з API конкретної платформи дослідження, як наприклад заміна реалізації класів обробника контролера на рівні апаратного забезпечення. Наприклад, для Windows була виконана наступна реалізація, яка наведена частково нижче (повний лістинг наведений у додатку з кодом у кінці записки).

**#**

**# C struct redefinitions**

**#**

**PUL = ctypes.POINTER(ctypes.c\_ulong)**

**class KeyBdInput(ctypes.Structure):**

**\_fields\_ = [**

**("wVk", ctypes.c\_ushort),**

**("wScan", ctypes.c\_ushort),**

**("dwFlags", ctypes.c\_ulong ),**

**("time", ctypes.c\_ulong ),**

**("dwExtraInfo", PUL )**

**]**

**class HardwareInput(ctypes.Structure):**

**\_fields\_ = [**

**("uMsg", ctypes.c\_ulong ),**

**("wParamL", ctypes.c\_short ),**

**("wParamH", ctypes.c\_ushort)**

**]**

**class MouseInput(ctypes.Structure):**

**\_fields\_ = [**

**("dx", ctypes.c\_long ),**

**("dy", ctypes.c\_long ),**

**("mouseData", ctypes.c\_ulong),**

**("dwFlags", ctypes.c\_ulong),**

**("time", ctypes.c\_ulong),**

**("dwExtraInfo", PUL )**

**]**

**def PressKey(\_hexKeyCode):**

**extra = ctypes.c\_ulong(0)**

**ii\_ = Input\_I()**

**ii\_.ki = KeyBdInput(0, \_hexKeyCode, 0x0008, 0, ctypes.pointer(extra))**

**x = Input(ctypes.c\_ulong(1), ii\_)**

**ctypes.windll.user32.SendInput(1, ctypes.pointer(x), ctypes.sizeof(x))**

Для визначення ліній полоси руху виявилось зручним використовувати концепцію регіону інтересу. Регіон інтересу (region of interest) – це така обрана підмножина зразків набору даних, визначена для конкретної мети. У конкретному випадку була використана наступна ділянка для визначення саме необхідних ліній маючи на увазі розташування датчику. Схематичне зображення подане на рисунку 2.7 нижче.

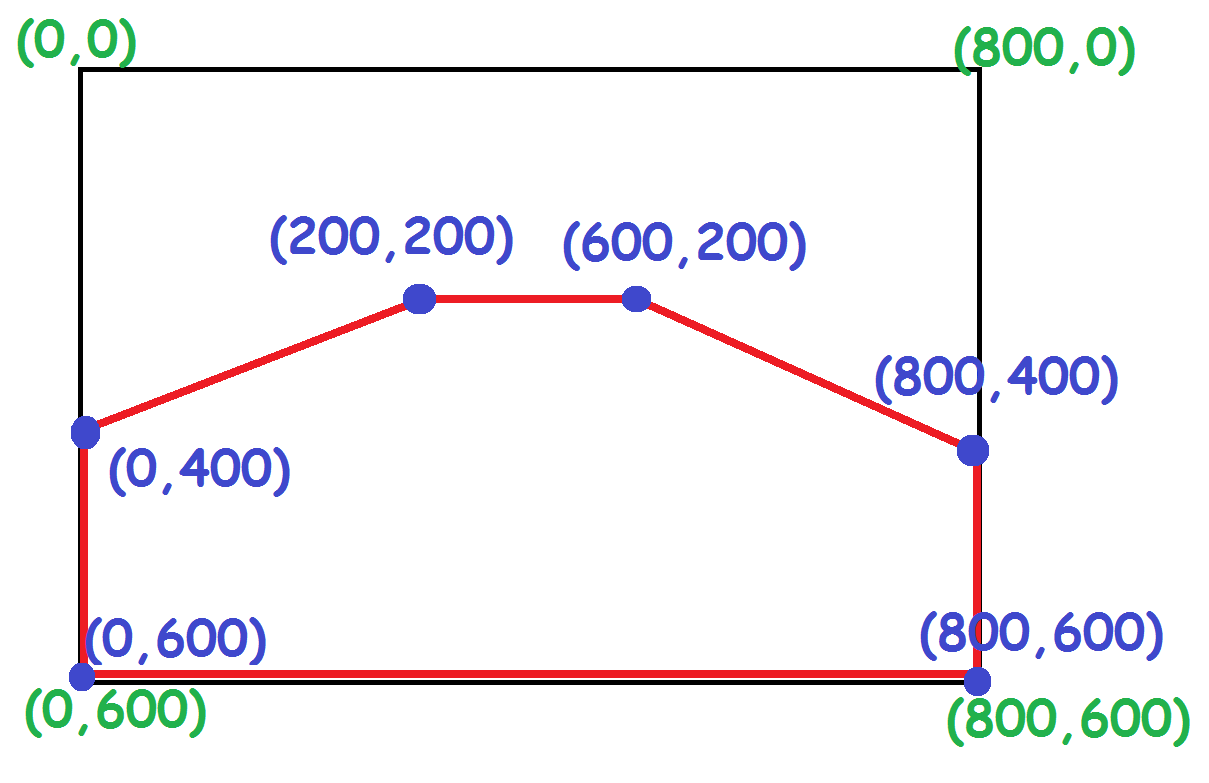


Рисунок 2.7 – Схематичне зображення обраного регіону інтересу

Дані координати обрані для вікна загальною розподільною здатністю вісімсот пікселів у ширину на шістсот пікселів у довжину. Для відкидання іншої частини зображення використовувалися методи бібліотеки OpenCV так як наведено нижче.

**def roi(\_image, \_vertices):**

**mask = np.zeros\_like(\_image)**

**cv2.fillPoly(mask, \_vertices, 255)**

**masked = cv2.bitwise\_and(\_image, mask)**

**return masked**

Загалом можна емпірично визначити регіон інтересу, якщо відомі розташування сенсорів напередодні проведення візуального аналізу отриманих даних. Для тренування використовувалась модель AxelNet, що була проініціалізована згідно запропонованому алгоритму наступним чином [23]:

**import tflearn**

**from tflearn.layers.conv import conv\_2d, max\_pool\_2d**

**from tflearn.layers.core import input\_data, dropout, fully\_connected**

**from tflearn.layers.estimator import regression**

**from tflearn.layers.normalization import local\_response\_normalization**

**def alexnet(width, height, lr):**

**network = input\_data(shape=[None, width, height, 1], name='input')**

**network = conv\_2d(network, 96, 11, strides=4, activation='relu')**

**network = max\_pool\_2d(network, 3, strides=2)**

**network = local\_response\_normalization(network)**

**network = conv\_2d(network, 256, 5, activation='relu')**

**network = max\_pool\_2d(network, 3, strides=2)**

**network = local\_response\_normalization(network)**

**network = conv\_2d(network, 384, 3, activation='relu')**

**network = conv\_2d(network, 384, 3, activation='relu')**

**network = conv\_2d(network, 256, 3, activation='relu')**

**network = max\_pool\_2d(network, 3, strides=2)**

**network = local\_response\_normalization(network)**

**network = fully\_connected(network, 4096, activation='tanh')**

**network = dropout(network, 0.5)**

**network = fully\_connected(network, 4096, activation='tanh')**

**network = dropout(network, 0.5)**

**network = fully\_connected(network, 3, activation='softmax')**

**network = regression(network, optimizer='momentum',**

**loss='categorical\_crossentropy',**

**learning\_rate=lr, name='targets')**

**model = tflearn.DNN(network, checkpoint\_path='model\_alexnet',**

**max\_checkpoints=1, tensorboard\_verbose=2, tensorboard\_dir='log')**

**return model**

# 3 ВИКОРИСТАННЯ ОТРИМАНИХ РЕЗУЛЬТАТІВ

ДОДАТИ ПРО НАСТУПНІ ЕТАПИ РОЗРОБКИ + ПРО МОЖЛИВІСТЬ ДОДАННЯ У АВТОМОБІЛЬ

ОГЛЯД НАУКОВОЇ ЛІТЕРАТУРИ

<http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>

<https://waymo.com/tech/> (Google car)

Автономна навігація за допомогою візуальних засобів є активною дослідницькою областю протягом багатьох років [1]. Одною з перших ідей для автономного автомобіля є відслідкування автомобілів та пересування зі іншими транспортними засобами, що є рутинною задачею під час міського трафіку, який складається з заторів та монотонного руху у завданній інфраструктурі міста. Іншою сферою застосування технологій допоміжника автоматичного пасивного пересування є рух по автомагістралі, якому також властиві якості монотонності та відносна залежність від пересування інших транспортниз агентів.

Автономні транспортні засоби знаходяться на межі сфер штучного інтелекту та інженерії. Неможливо досягти результату не використовуючи комплексний підхід роботи з інженерними досягненнями та використанням останніх досліджень у області аналізу та прийняття рішень.

Сучасна парадигма автомобільної індустрії зсунула свої пріорітети з потужних високоефективних автомобілей у сторону комфорту та безпеки як головних критеріїв конструювання. Цей зсув парадигми надав прискорення розробці різноманітних розумних технологій у засобах руху. Ультимативним рішення проблеми максимізації комфорту та безпеки є розробка автономного автомобіля. Для виконання цієї міссії автомобіль повинен аналізувати свою середу пересування, розробляти план пересування та безпосередньо керування без втручання людини.

Такі організації як Defensive Advanced Research Project Agency відкривають змагання для автономних автомобілей у пересічній місцевості та в урбаністичному оточенні. Перше змагання зосереджено на досягненні мети автономного пересування за умовою неможливості безпосереднього контролю людиною у реальному часі. Наприклад, ровер на Марсі або не потребуючий контролю розміновувач у військовому середовищі тощо. Друге змагання має на меті впровадження технологічних ноу-хау та досліджень у серійні автомобілі задля підвищення безпеки та комфорту пересування в урбаністичних умовах. У результаті глобальні автомобільні компанії постійно інвестують у розвиток коммерціалізації автономних транспортних засобів.

Одною з пріорітетних задач урбаністичної системи автономного автомобілю є відслідкування статичних перешкод (стіни, ліхтарі, паркани та інше) та динамінчих перешкод. Друга категорія більш широка і в неї потрапляють такі об’єкти як пішохіди, тварини, інші транспортні агенти (автомобілі, мотоцикли, велосипеди та інше). Також важливою темою для автоматизації руху є пересування у межах дорожніх правил, що включає у себе світлофори, різного роду дорожні знаки, пішоходні переходи, регулювальника та інші допоміжні сигнальні засоби.

Розробка такого транспортного засобу інтегрує технології з двух сфер людської діяльності: автомобільна індустрія та промисловість роботів. Надійну механічну та електричну платформу для автономних автомобілей можна досягти лише за допомогою продуктів діяльності автомобільної індустрії. Багато алгоритмів автономного пересування було досліджено

(<http://ieeexplore.ieee.org/abstract/document/6809196/>)

ПЕРЕЛІК ПОСИЛАНЬ

[1] Jeong E and Oh C 2017 Evaluating the effectiveness of active vehicle safety systems Accident Analysis & Prevention 85-96

[2] Ni L, Gupta A, Falcone P and Johannesson L 2016 Vehicle lateral motion control with perfomance and safety guarantees IFAC Proceedings Volumes 285-90

[3] Bakfish K and Hajnc D 2003 New book about tires (Moscow: Izdatel'stvo Astrel') – in Russian

[4] Buznikov S E 2009 The method of constructing information virtual sensors for car’s active safety systems Proc. of XVII Int. Conf. “The management problems of safety in complex systems” (Moscow: Russian State University for the Humanities Press) pp 420-4 – in Russian

[5] Buznikov S E, Elkin D S, Shabanov N S and Strukov V O 2016 Task of safe automatic braking of the vehicle Trudy NAMI 44-52 – in Russian

[6] Buznikov S E, Elkin D S 2007 Identification of maximal values of sliding friction coefficients of the vehicle wheels: the certificate of official registration program for computer # 2007610818 Rospatent

[7] Saykin A, Bakhmutov S, Terenchenko A, Endachev D, Karpukhin K and Zarubkin V 2014 Tendency of Creation of "Driverles" Vehicles Abroad Biosciences Biotechnology Research Asia 11 p 241-6

[8] Saikin A, Buznikov S and Karpukhin K 2016 The Analysis of Technical Vision Problems Typical for Driverless Vehicles Research Journal of Pharmaceutical, Biological and Chemical Sciences 7 #4 p 2053-9

[9] Shadrin S, Ivanov A and Karpukhin K 2016 Using Data From Multiplex Networks on Vehicles in Road Tests, in Intelligent Transportation Systems, and in Self-Driving Cars Russian Engineering Research 36 #10 p 811-4 doi:10.3103/S1068798X16100166

[10] Dakroub H, Shaout A and Awajan A 2016. Connected Car Architecture and Virtualization SAE Int. J. Passeng. Cars – Electron. Electr. Syst. 9(1) p 153-9 doi: 10.4271/2016-01-0081

[11] Shadrin S and Ivanov A 2016 Algorithm of autonomous vehicle steering system control law estimation while the desired trajectory driving ARPN Journal of Engineering and Applied Sciences 11 #15 p 9312-6