Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *Єфременка Дмитра Валерійовича*
(ПІБ)

академічної групи *121-18ск-1*
(шифр)

спеціальності *121 Інженерія програмного забезпечення*
(код і назва спеціальності)

освітньої програми *Інженерія програмного забезпечення*
(назва освітньої програми)

на тему: *Розробка програмного забезпечення на мові Python для виявлення сонливості водія*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*проф. Корнієнко В.І.*			
розділів:				
спеціальний	*проф. Корнієнко В.І.*			
економічний	*проф. Вагонова О.Г.*			
Рецензент	*доц. Шедловський І.А.*			
Нормоконтролер	*доц. Гуліна І.Г.*			

Дніпро
2021

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 2021 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *121-18ск-1* (група) *Єфременка Дмитра Валерійовича* (прізвище та ініціали)

тема кваліфікаційної роботи *Розробка програмного забезпечення на мові Python для виявлення сонливості водія*

затверджена наказом ректора НТУ «ДП» від «07» червня 2021 р. № 317-с

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів виробничої практики та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі.*** ***Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***13.05.2021 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПО й тривалості його розробки***	***27.05.2021р.***

Завдання видав _____ *проф. КорнієнкоВ.І.*
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Єфременко Д.В.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *14.01.2021 р.*

Термін подання кваліфікаційної роботи до ЕК: *10.06.2021 р.*

РЕФЕРАТ

Пояснювальна записка: 60 с., 10 рис., 1 табл., 3 дод., 23 джерела.

Об'єкт розробки: система виявлення сонливості водія.

Мета кваліфікаційної роботи: створити систему виявлення сонливості, яка буде визначати, що очі людини закриті на кілька секунд. Ця система попередить водія про небезпеку при виявленні сонливості.

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета кваліфікаційної роботи та галузь її застосування, наведено обґрунтування актуальності теми та уточняється постановка завдання.

У першому розділі проведено аналіз предметної галузі, визначено актуальність завдання та призначення розробки, розроблена постановка завдання, задані вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі виконано аналіз існуючих рішень, обрано платформу для розробки, виконано проектування і розробка програми, наведено опис алгоритму і структури функціонування програми, визначені вхідні і вихідні дані, наведені характеристики складу параметрів технічних засобів, описаний виклик та завантаження програми, описана робота програми.

В економічному розділі визначено трудомісткість розробленого програмного забезпечення, проведений підрахунок вартості роботи по створенню програми та розраховано час на його створення.

Практичне значення полягає у створенні системи виявлення сонливості, яка буде визначати, що очі людини закриті протягом заданого часу.

Актуальність даного програмного продукту визначається великим попитом на подібні розробки, що не перенавантажені комерційними елементам, такі як: реклама, програми дистрибуції та інші.

Список ключових слів: ADAS-СИСТЕМИ, СОНЛИВІСТЬ ВОДІЯ, ТЕХНОЛОГІЇ БЕЗПЕКИ ВОДІЯ.

ABSTRACT

Explanatory note: 57 pp., 12 figs., 1 tab., 3 appendices, 23 sources.

Object of development: system for detecting driver drowsiness.

The purpose of the qualification work: to create a system for detecting drowsiness, which will determine that a person's eyes are closed for a few seconds. This system will warn the driver of the danger of drowsiness.

The introduction considers the analysis and the current state of the problem, specifies the purpose of the qualification work and the field of its application, provides a justification for the relevance of the topic and clarifies the problem.

In the first section the analysis of the subject area is carried out, the urgency of the task and purpose of development is defined, the statement of the task is developed, requirements to software realization, technologies and software are set.

The second section analyzes existing solutions, selects a platform for development, performs design and development of the program, describes the algorithm and structure of the program, identifies input and output data, provides characteristics of the parameters of hardware, describes the call and download of the program, describes the program.

In the economic section, the complexity of the developed software is determined, the cost of work on creating the program is calculated and the time for its creation is calculated.

The practical significance is to create a system for detecting drowsiness, which will determine that a person's eyes are closed for a specified time.

The relevance of this software product is determined by the high demand for such developments that are not overloaded with commercial elements, such as: advertising, distribution programs and others.

Keywords: ADAS SYSTEMS, DRIVER DRIVING, DRIVER SAFETY TECHNOLOGIES.

ЗМІСТ

СПИСОК УМОВНИХ ПОЗНАЧЕНЬ

ADAS – Advanced Driver Assistance Systems (просунуті системи допомоги водієві);

API – Application Program Interface;

ОС – операційна система;

ПК – персональний комп’ютер;

ТЗ – транспортний засіб.

ВСТУП

Кожна людина чула про таку проблему, як сонний водій. Але чомусь втома або сонливість в процесі водіння автомобіля не здаються серйозними проблемами.

Коли водій сідає за кермо автомобіля, не виспавшись, він наражає на небезпеку себе і всіх на дорозі. Якщо водій не спав більше двадцяти годин, його стан можна порівняти зі станом нетверезого водія, оскільки знижується реакція, здатність керувати автомобілем, порушується координація руху, водій допускає більше помилок. Сонливе й стомлене водіння є серйозною проблемою безпеки на дорозі, а також є причиною тисяч нещасних і навіть смертельних випадків щороку. Дорожні аварії, як правило, дуже серйозні через значну втрату уваги, що часто призводить до непередбачуваної траєкторії руху транспортного засобу і відсутності реакції на гальмування. Для запобігання таких аварій необхідні надійні системи безпеки – системи виявлення сонливості водія. Це технологія безпеки автомобіля, яка допомагає запобігти аваріям, викликаним сонливістю водія, отже тема роботи є актуальною.

Об’єкт дослідження: сонливість під час водіння транспортного засобу.

Предмет дослідження: зменшення наслідків сонливості під час водіння транспортного засобу.

Мета роботи: створити систему виявлення сонливості та попередження водія про небезпеку.

Завдання:

- проаналізувати сучасний стан проблеми сонливості за кермом;
- порівняти наявні на ринку продукти для усунення наслідків сонливості за кермом;
- сформулювати функціональні вимоги до програмного забезпечення;
- спроєктувати алгоритми та структури даних;
- створити програмну реалізацію та здійснити її тестування.

Для більшості користувачів ця система зможе стати доступною і, що найголовніше, надійною системою додаткової безпеки під час поїздки.

Завдання кваліфікаційної роботи та об'єкт його діяльності безпосередньо пов'язані зі спеціальністю 121 «Інженерія програмного забезпечення» та відповідає узагальненій тематиці кваліфікаційних робіт і переліку зазначених компетенцій.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Загальні відомості з предметної галузі

Перевтома або сон за кермом реєструються як причина незначної частки ДТП в Україні (80-160 випадків за рік) [1], що не може відповідати реаліям. В одному з досліджень [2] було продемонстровано, що серед водіїв вантажівок, зайнятих на перевезеннях у нічний час, понад 37% мали порушення сну, 12.5% страждали на важкі форми синдрому сонних апное (ССА).

Серед водіїв, які страждали на порушення сну, 2/3 відмічали епізоди сну за кермом, 42% потрапляли у ДТП, які вони пов’язували з підвищеною втомленістю. Разом з цим, 95.5% водіїв не визнавали зв'язку між розладами сну і ДТП у нічний час.

Втомлюваність має прояви у вигляді статичного перенапруження опорно-рухового апарату, пригнічення сенсорно-моторних реакцій, зниження пам’яті й стійкості до монотонності, роздратованості й підвищення агресивності. Понад 25% водіїв відзначають регулярні епізоди засинання за кермом. В експерименті, завдяки реєстрації рухів очима професійних водіїв під час керування вантажівками, було встановлено, що у 14% випадків керування у водіїв реєструвалося не менше двох епізодів заплющених очей понад 3 сек. Втомлюваність і сонливість за кермом за даними різних авторів є причиною 15-25% всіх дорожніх пригод.

Системи автоматизованої підтримки водія (Advanced Driver Assistance Systems, ADAS) націлені на надання допомоги водієві в справі недопущення дорожньо-транспортних пригод (ДТП) або пом'якшення їх наслідків. Попереджувальні сигнали високої пріоритетності подаються цими системами для стимулювання пильності і своєчасних і належних дій водія в ситуаціях, коли може мати місце або безпосередньо існує небезпека виникнення серйозних ушкоджень або загибелі людей.

Основні сучасні технології, які складають ADAS-системи:

– система виявлення дорожньої розмітки (Lane Detection, LDA)– маркерів смуг і краю проїжджої частини, оцінка стану автомобіля в межах смуги;

– система контролю рядності руху (Lane Departure Warning, LDW) використовує інформацію від модуля виявлення смуги LDA, обчислює час до перетину розмітки (Time to Lane Crossing, TLC) і забезпечує попередження водієві в разі виявлення догляду;

– функція виявлення транспортних засобів (ТЗ), що працює на основі монокамерних алгоритмів, розпізнає всі моторизовані механічні ТЗ–автомобілі, мотоцикли, вантажівки, в умовах денного або нічного освітлення;

– функція попередження про передньому зіткненні і пом'якшення неминучої аварії (Forward Collision Warning and pre-crash mitigation), шосейний моніторинг і попередження (Headway Monitoring and Warning).

Всі ADAS-системи можна умовно розділити за формою подання на дві категорії:

– мобільні додатки, що встановлюються на смартфон з магазину додатків (Google Play / App Store);

– зовнішні камери, сенсори, датчики, чіпи і т.д. разом з програмними засобами, що вбудовуються в автомобілі на заводах-виробниках або встановлюються постфактум.

При аналізі інформації про наявні на ринку аналоги, з’ясувалось, що кожна автомобільна компанія створює свою, спеціально адаптовану модель цієї системи для своїх автомобілів. Є різні технології відстеження сонливості водія: моніторинг моделі рульового управління, моніторинг положення автомобіля відносно смуги руху, контроль очей/обличчя водія, фізіологічні виміри. І кожну з цих технологій компанії використовують в своїх автомобілях або поодинці, або комбінуючи їх.

Таблиця 1.1.

Порівняння систем виявлення сонливості водія

Назва бренда	Назва системи	Критерії виявлення сонливості	Спосіб попередження водія	Перенесимість на інші автомобілі
Renault	Fatigue Detection Warning	- рухи керма; - дії водія та інших пристроїв(вказівників повороту, двірників і т.д.); - час, проведений за кермом без зупинок.	Звуковий та візуальний сигнали	Відсутня
Kia	Driver Attention Warning (DAW)	-стиль водіння; -поведінка водія.	Звуковий та візуальний сигнали	Відсутня
Skoda	IBuzz Alert	-рухи керма	Звуковий та візуальний сигнали	Відсутня

На ринку ADAS-систем існує чимала кількість рішень від різних компаній. Можна відзначити наступні недоліки інтегрованих ADAS-систем:

– вартість таких систем залишається досить високою;

– вони доступні в основному тільки у вигляді додаткової опції для дорогих і ексклюзивних автомобілів.

Одним з найбільш популярних мобільних додатків в сфері ADAS-систем є рішення iOnRoad3[15]. Додаток використовує вбудовані в смартфон тилову камеру, сенсори і датчики для виявлення попереду автомобілів, що йдуть і попередження водія у разі небезпеки (рис. 1.1. – 1.3.). Дана система фіксує об'єкти перед водієм в реальному часі, вираховуючи поточну швидкість за допомогою сенсорів. При насувається спливає звукове та графічне

попередження про можливість зіткнення, дозволяючи водієві вчасно загальмувати.

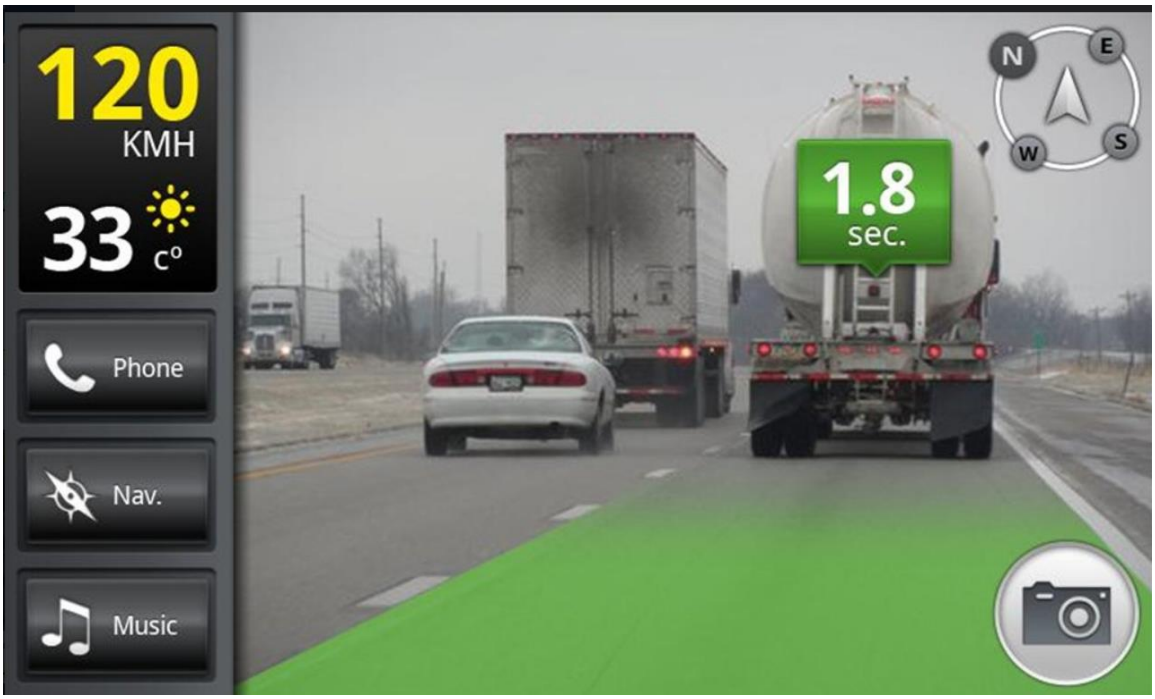

Рис. 1.1. Зображення роботи системи iOnRoad3

Рис. 1.2. Зображення роботи системи iOnRoad3

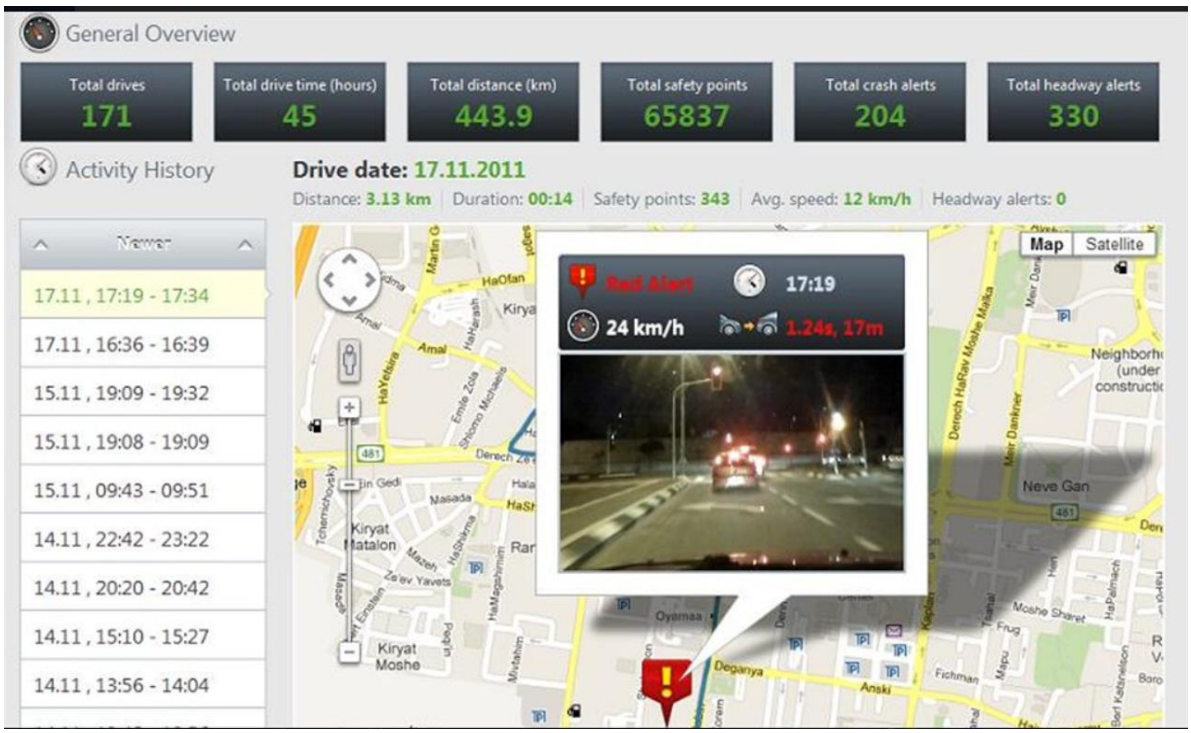

Рис. 1.3. Зображення роботи системи iOnRoad3

Мобільні додатки мають, в свою чергу, свої недоліки. Головним з них є те, що мобільні ADAS-рішення не вміють відстежувати одночасно і поведінку водія, і рух транспортних засобів, і в таких додатках задіюється тільки одна з наявних на смартфоні камер. Таким чином, мобільні ADAS-рішення не враховують весь спектр небезпечних дорожніх ситуацій, з якими може зіткнутися водій ТЗ. Крім того, в таких мобільних рішеннях все інтенсивні обчислення з обробки та аналізу зображень та дорожніх ситуацій виконуються тільки на смартфоні водія, що може бути не завжди прийнятно через обмеженість ресурсів мобільних пристроїв. Таким чином, розгортання і виконання процесів мобільних додатків не представлено в хмарному середовищі.

Головна відмінність даної реалізації буде полягати в тому, що вона написана мовою програмування Python, а тому зможе вподальшому запускатися на будь-якому пристрої, чи то вбудованому комп'ютері автомобіля, чи смартфоні на Android або iOS. Це зробить програму вкрай доступною та привабливою у майбутньому. Всі названі вище системи можуть запускатися

лише на конкретному комп'ютері від конкретного виробника автомобіля, під який вони були розроблені. Про запуск на смартфоні навіть не доводиться говорити.

Впровадження систем комп'ютерного зору дуже залежить від області їх застосування. Деякі системи є автономними і вирішують специфічні проблеми детектування та вимірювання, тоді як інші системи складають підсистеми більших систем, які, наприклад, можуть містити підсистеми контролю за механічними маніпуляторами, планування, інформаційні бази даних, інтерфейси людина-машина тощо. Реалізація систем комп'ютерного зору також залежить від того, чи є її функціональність наперед визначеною чи деякі її частини можуть бути вивчені і удосконалені в процесі роботи. Однак, існують функції, типові для багатьох систем комп'ютерного зору.

Отримання зображень: цифрові зображення отримуються від одного чи декількох датчиків зображення, які окрім різноманітних типів світлочутливих камер мають давачі 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.

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:

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

– видалення шумів задля того, щоби видалити спотворення, які вносяться давачем;

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

– масштабування для кращого розрізнення структур на зображенні.

Виокремлення деталей: деталі зображення різного рівня складності виділяються з відеоданих. Типовими прикладами таких деталей є:

– лінії та межі;

– локалізовані точки інтересу, такі як кути, краплі чи точки: складніші деталі можуть належати до структури, форми чи руху.

Детектування/Сегментація: на певному етапі обробки приймається рішення про те, які точки чи ділянки зображення є важливими для подальшої обробки. Прикладами є: виділення визначеного набору точок, що нас цікавлять.

Сегментація одного або кількох ділянок зображення, які містять характерний об'єкт.

Високорівнева обробка: на цьому кроці вхідні дані зазвичай представляють невеликий набір даних, наприклад, набір точок чи ділянка зображення, в якій за припущенням знаходиться певний об'єкт.

Прикладами є:

– перевірка того, що дані задовольняють умовам, що залежать від методу і застосування;

– оцінка характерних параметрів, таких як положення або розмір об'єкта;

– класифікація знайденого об'єкта за різними категоріями.

Якщо розглядати завдання моніторингу та контролю втоми водія, то її можна розділити на три підзадачі – виявлення, розпізнавання і відстеження.

Виявлення (detection) – виділення областей на зображенні, які можуть містити цікавий для нас об'єкти, в нашому випадку особа водія.

Розпізнавання (recognition) – уточнення типів знайдених об'єктів, а саме очі водія.

Супровід (tracking)–локалізація на наступних кадрах розпізнаних об'єктів.

Існує два способи відстеження особи водія на фото. Перший – виявлення об'єкта в кожному кадрі і другий – супровід через виявлення в першому кадрі, тобто виділяється об'єкт, розпізнається і надалі відслідковується на кожному наступному кадрі. Даний спосіб є обчислювально ефективним на відміну від першого, де доводиться шукати об'єкти в кожному кадрі, що веде до суттєвої обчислювального навантаження на систему [15].

Виходячи з вище проаналізованого, система сонливості водія наразі є актуальною, та, оскільки аналогів в Україні немає, було поставлено завдання розробити повнофункціональний прототип програми для виявлення сонливості водія.

1.2. Призначення розробки та галузь застосування

Основним призначенням розробки є створення системи виявлення сонливості, яка буде визначати, що очі людини закриті на кілька секунд. Ця система зможе попередити водія про небезпеку при виявленні сонливості. Оскільки водії страждають від нестачі сну, то через що стає дуже небезпечно керувати автомобілем в такому стані. Тому більшість аварій відбувається через сонливості водія.

Галуззю застосування даної розробки може бути виробництво ситем безпеки транспортних засобів та водія, а також результати розробки можна вподальшому використати в різних галузях, де є необхідність відстеження соливості людини, наприклад при виконанні нею службових обов’язків (чергування, управляння технологічним процесом, тощо).

1.3. Підстави для розробки

Відповідно до освітньої програми, згідно навчального плану та графіків навчального процесу, в кінці навчання студент виконує кваліфікаційну роботу.

Отже, підставами для розробки (виконання кваліфікаційної роботи) є:

- освітня програма спеціальності 121 «Інженерія програмного забезпечення»;
- навчальний план та графік навчального процесу;
- наказ ректора Національного технічного університету «Дніпровська політехніка» № 317-с від 07.06.2021 р;
- Завдання для розробки кваліфікаційної роботи на тему «Розробка програмного забезпечення на мові Python для виявлення сонливості водія».

1.4. Постановка завдання

Завдання:

– проаналізувати сучасний стан проблеми сонливості за кермом;

– порівняти наявні на ринку продукти для усунення наслідків сонливості за кермом;

– сформулювати функціональні вимоги до програмного забезпечення;

– спроєктувати алгоритми та структури даних;

– створити програмну реалізацію та здійснити її тестування.

Висунуто вимоги до самої програми:

- програма має бути простою і надійною у використанні;
- вона повинна мати можливість інтеграції в наявну систему транспорту;
- повинна мати зручний та інтуітивно зрозумілий інтерфейс;
- повинна бути невимогливою до програмно-апаратних засобів.

1.5. Вимоги до програми або програмного виробу

1.5.1. Вимоги до функціональних характеристик

Висунуто вимоги до самої програми:

- програма має бути простою і надійною у використанні;
- вона повинна мати можливість інтеграції в наявну систему транспорту;

- повинна мати зручний та інтуітивно зрозумілий інтерфейс;
- повинна бути невимогливою до програмно-апаратних засобів.

1.5.2. Вимоги до інформаційної безпеки

Оскільки додаток не повинен довгостроково зберігати дані, а їх зміст не є конференційною інформацією, то специфічних вимог до інформаційної безпеки немає.

Достатніми будуть встановлені в системі програмні та організаційні засоби забезпечення захисту інформації.

1.5.3. Вимоги до складу та параметрів технічних засобів

Оскільки розроблене програмне забезпечення планується як універсальне, що може бути використане на різних типах пристроїв (ПК, ноутбук, планшет, смартфон чи бортовик комп’ютер автомобіля), то одним з критеріїв є невимогливість до програмно-апаратних засобів.

Додаток повинен виконуватись на мобільних пристроях під керівництвом різних операційних систем. Мінімальна діагональ екрану для пристрою повинна бути не менш ніж 5 дюйми з щільністю точок не менш ніж 480×800 dpi. для зручного використання

Програма повинна бути адаптована для роботи й на планшетах, а також телефонах з великим розширенням і великою діагоналлю екрану. Зміст програми при цьому не повинен втратити своїх якостей читаності.

Також слід зазначити, що головною вимогою до складу та параметрів технічних засобів є наявніть камери (рекомендовано з високою якістю) та наявністтю функції зйомки вночі.

1.5.4. Вимоги до інформаційної та програмної сумісності

Додаток не є вимогливим до інформаційної та програмної сумісності та повивен підтримувати роботу на різних пристроях в тому числі й мобільних пристроях різних конфігурацій під управлінням різних ОС.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

2.1. Функціональне призначення програми

Основним призначенням розробки є створення системи виявлення сонливості, яка буде визначати, що очі людини закриті на кілька секунд. Ця система зможе попередити водія про небезпеку при виявленні сонливості. Оскільки водії страждають від нестачі сну, то через що стає дуже небезпечно керувати автомобілем в такому стані. Тому більшість аварій відбувається через сонливості водія.

Галуззю застосування даної розробки може бути виробництво ситем безпеки транспортних засобів та водія, а також результати розробки можна вподальшому використати в різних галузях, де є необхідність відстеження соливості людини, наприклад при виконанні нею службових обов’язків (чергування, управляння технологічним процесом, тощо).

Таким чином, можна сказати, що основною функцією програмого продукту є повідомлення про заплющенні очі у суб’єкта спостереження.

2.2. Опис застосованих математичних методів

В ході розробки додатку застосовувались методи та моделі для аналізу зображення та навчання програми, інші математичні методи не застосовувалися. Відповідні алгоритми і функції, що викоритовують математичні методи є стандартними і знаходяться у реалізаціях відповідного програмного забезпечення, що зазначено в п. 2.4.

2.3. Опис використаної архітектури та шаблонів проектування

У розробці ПЗ часто зустрічаються проблеми, які вже розв'язувалися раніше в інших проектах. У зв'язку з тим, що контексти, в яких дана проблема розв'язувалась, можуть різнитися – (інший тип додатку, інша платформа або інша мова програмування), – все зазвичай закінчується повторенням проектування і реалізації даного розв'язку, – тим самим виникає ситуація «повторного винаходу колеса».

Шаблони дозволяють базуватися на колективному досвіді кваліфікованих інженерів по проектуванню. Вони фіксують існуючий досвід розробки, що добре себе зарекомендував. Кожен шаблон має справу з конкретною проблемою, що багато разів зустрічається в області проектування і реалізації. Шаблон – це опис добре перевіреної, узагальненої схеми розв'язку деякої проблеми (задачі), що часто повторюється під час розробки ПЗ, яка виникає в деяких специфічних умовах (контексті). Схема розв'язку проблеми задається шляхом – визначення використовуваних (складових): компонент, їх відповідальностей та способів їх взаємодії.

Властивості шаблонів:

1. Шаблони описують розв'язок для задач проектування, що часто повторюються, які виникають в деяких специфічних ситуаціях.

2. Шаблони документують накоплений досвід проектування, що добре себе зарекомендував.

3. Шаблони визначають і описують абстракції, які знаходяться на вищому рівні, ніж рівень окремих класів і екземплярів або компонентів.

4. Шаблони надають спільний словник термінів і загальне розуміння принципів проектування.

Шаблони архітектури ПЗ описують базові схеми структурної організації програмних систем, надають набір наперед визначених підсистем, визначають їх відповідальності, та включають правила і рекомендації по організації взаємодії між ними.

Види архітектурних шаблонів: Layers (рівні), Pipes and Filters (канали і фільтри), Blackboard (інформаційна "дошка"), Broker (брокер), Model-View-Controller (Модель-Представлення-Контролер), Presentation-Abstraction-Control (Представлення-Абстракція-Контролер), Microkernel (мікроядро), Reflection (відображення).

Оскільки сам код проекту не є великим за об’ємом та не є наповненим великим функціоналом, а сам проект планувався як універсальний та кросплатформній, то при розробці було використано шаблон проектування Model - View - Controller (Модель – Представлення - Контролер)

2.4. Опис використаних технологій та мов програмування

Для проєкта потрібні веб-камера, з якої ми будемо отримувати зображення, мова програмування Python (рекомендується версія 3.6), а потім за допомогою pip треба встановити необхідні пакети:

1. OpenCV [3] - pip install opencv-python (виявлення обличя и очей).
2. TensorFlow [4] - pip install tensorflow (keras використовує TensorFlow як бекенд).
3. Keras [5] - pip install keras (для побудови нашої класифікаційної моделі).
4. Pygame [6] - pip install pygame (для відтворення звукового сигналу).

Для написання коду використовувався редактор тексту Sublime Text 3, оскіли він є простий і інтуїтивно зрозумілий у використанні, а також підтримує плагіни для мови Python.

Python проста у використанні, та водночас повноцінна мова програмування, що надає набагато більше засобів для структурування і підтримки великих програм, ніж shell. З іншого боку, вона краще за C обробляє помилки, і, будучи мовою дуже високого рівня, має вбудовані типи даних високого рівня, такі як гнучкі масиви і словники, ефективна реалізація яких на C потребує значних витрат часу.

Завдяки більш загальним типам даних, Python застосовують до більш широкого кола задач, ніж Awk і навіть Perl, у той ж час багато речей на мові Python робляться настільки ж просто.

Python дозволяє розбивати програми на модулі, що потім можуть бути використані в інших програмах. Python поставляється з великою бібліотекою стандартних модулів, які можна використовувати як основу для нових програм або як приклади при вивченні мови. Стандартні модулі надають засоби для роботи з файлами, системними викликами, мережними з'єднаннями і навіть інтерфейсами до різних графічних бібліотек.

Python - інтерпретована мова, що дозволяє заощадити значну кількість часу, що зазвичай витрачається на компіляцію. Інтерпретатор можна використовувати інтерактивно, що дозволяє експериментувати з можливостями мови, писати шаблони програм або тестувати функції при розробці “знизу-вверх”. Він також зручний як настільний калькулятор. Python дозволяє писати дуже компактні й зручні для читання програми. Програми, написані мовою Python, звичайно значно коротші еквівалента на C або C++ з декількох причин:

– типи даних високого рівня дозволять Вам виразити складні операції однією інструкцією;

– групування інструкцій виконується за допомогою відступів замість фігурних дужок;

– немає необхідності в оголошенні змінних.

Python розширювана мова: знання C дозволяє додавати нові функції, що вбудовуються, або модулі для виконання критичних операцій з максимальною швидкістю або написання інтерфейсу до комерційних бібліотек, доступним тільки у двійковій формі. Інтерпретатор мови Python може бути вбудований у програму, написану на C, і використовувати його як розширення або командну мову для цієї програми. Python використовується в даний час десятками тисяч програмістів в усьому світі, і число людей, що використовують його, швидко зростає, подвоюється і потроюється щороку. Python приваблює користувачів з ряду причин. Він використовується для розробки програм і дозволяє провести

розробку набагато швидше, ніж традиційні мови типу C, C++ або Java . Ця мова працює однаково добре на Windows, UNIX, Macintosh, і OS/2, може використовуватися, для легкої розробки як малих додатків чи сценаріїв, так і для розгортання великих програм. Python пропонує доступ до могутнього і легкого у використанні комплекту 29 інструментальних засобів графічного інтерфейсу користувача. Традиційні машинні мови типу C і Pascal мають ряд характеристик, наприклад, сувора типізація, базові типи, складні (і звичайно довгі) цикли, і потреба у великих кількостях кодів для виконання відносно малих задач. Java досить новий, але розділяє більшість характеристик, включених у цей перелік. Програмісти, знайомі з традиційними мовами погодяться, що відсутність суворої типізації полегшує роботу з Python.

Відмінностей Python від інших мов доволі багато, перерахуємо основні з них:

– Керування пам'яттю - цілком автоматичне — не потрібно хвилюватися щодо розподілу або звільнення пам'яті. Немає загрози “небезпечного посилання”. Java - єдина мова, що пропонує таку концепцію.

– Типи зв'язані з об'єктами, а не зі змінними. Це означає, що змінній може бути призначене значення будь-якого типу, і що (наприклад) масив може містити об'єкти різних типів. Традиційні мови не надають такої можливості.

– Операції звичайно виконуються в більш високому рівні абстракції. Це частково результат того, як написана мова, і частково результат розширеної стандартної бібліотеки кодів, що поставляється разом з Python.

Ці та інші особливості Python роблять розгортання додатків надзвичайно швидким. Продуктивність створеного додатку залежить від його особливостей. Звичайно, для чисельного алгоритму, що виконує звичайну арифметику цілого числа в циклі 'for', неважливо, на якій мові він написаний. Але для “середнього” додатка, збільшення продуктивності може бути просто дивовижним. Один недолік Python, у порівнянні з найбільш традиційними мовами, полягає в тому, що це - не цілком компільована мова; замість цього, вона частково транслює програму до внутрішньої форми байт-коду, і цей байт-код виконується

інтерпретатором Python. Однак, у перспективі – сучасні комп'ютери мають так багато невикористовуваного обчислювального потенціалу, що для 90% додатків швидкодія зв'язана з вибором мови. Java теж компілюється в байт-код, але в даний час працює повільніше ніж Python у більшості випадків. Крім того, дуже просто об'єднати Python з модулями, написаними на C або C++, які можна використовувати, щоб збільшити швидкість роботи програм в критичних ділянках.

OpenCV - це кроссплатформена бібліотека, за допомогою якої можна розробляти програми для комп'ютерного зору в реальному часі. Основна увага приділяється обробці зображень, захоплення і аналізу відео, включаючи такі функції, як виявлення осіб і виявлення об'єктів.

Computer Vision може бути визначена як дисципліна, яка пояснює, як реконструювати, переривати і розуміти тривимірну сцену з її 2D-зображень з точки зору властивостей структури, яка присутня в сцені. Він займається моделюванням і копіюванням людського зору з використанням комп'ютерного програмного і апаратного забезпечення.

Computer Vision значно перекриває такі поля:

Обробка зображень - фокусується на маніпулюванні зображеннями.

Розпізнавання образів - пояснює різні методи класифікації зразків.

Фотограмметрія - це стосується отримання точних вимірювань із зображень.

Computer Vision Vs Обробка зображень. Обробка зображень пов'язана з перетворенням зображення в зображення. Вхід і вихід обробки зображення обидва зображення.

Комп'ютерний зір - це побудова явних, які є значущими описів фізичних об'єктів по їх зображенню. Результатом комп'ютерного зору є опис або інтерпретація структур в тривимірній сцені.

OpenCV — бібліотека функцій та алгоритмів з відкритим кодом, призначена для обробки зображень і чисельних алгоритмів загального призначення на основі комп’ютерного зору Бібліотека була розроблена у 1999

році компанією Intel (Intel Research) з метою розвивати додатки, які навантажують CPU. На початкових етапах розробники ставили перед собою такі задачі:

- забезпечити розвиток технологій у сфері комп'ютерного зору
- створити бібліотеку з добре оптимізованим і відкритим кодом
- створити умови для розвитку комерційних додатків на основі комп'ютерного зору

На сьогоднішній час підтримкою і подальшою розробкою займається Willow Garage та Itseez. Бібліотека реалізована на C / C++ і також доступні API для мов програмування Python, Java, Ruby, Matlab, Lua та інших. Спочатку OpenCV розроблялась C, тому досі є інтерфейс для цієї мови програмування. На даний момент існують також обгортки для OpenCV для мов програмування C#, CH, Perl, Haskell і Ruby. Ці обгортки були розроблені для того, щоб збільшити аудиторію. Розвиток бібліотеки продовжується на мові програмування C++. Так як проект з відкритим кодом, то участь у розробці проекту може брати кожен охочий. OpenCV підтримує такі операційні системи:

- Windows;
- Linux;
- MacOS;
- FreeBSD;
- NetBSD;
- OpenBSD 24;
- Android;
- IOS;
- Maemo;
- BlackBerry.

На сьогоднішній день основний напрямок розвитку бібліотеки — інтеграція машинного навчання, для покращення результатів роботи. Уже є підтримка таких фреймворків:

- TensorFlow (розроблений компанією Google);

▪ PyTorch (розробленийй компанією Facebook);

▪ Caffe (розроблений науковцями з Університету Каліфорнії).

TensorFlow - відкритий Python-фреймворк машинного навчання (Machine Learning), створений командою Google Brain, об'єднує безліч моделей і алгоритмів глибокого навчання (Deep Learning). TensorFlow надає API-інтерфейс на Python, а самі обчислення відбуваються на високопродуктивному C ++.

Даний фреймворк може навчати глибокі нейронні мережі для класифікації рукописних цифр, розпізнавання зображень, вбудовування слів, рекурентних нейронних мереж, моделей для машинного перекладу, обробки природної мови (NLP) і т.д. Всі моделі машинного навчання можуть бути випущені в prdouction за допомогою інструменту TensorFlow Serving.

На даний момент сучасною версією є TensorFlow 2. Але деякі Data Scientist'и досі використовують 1-ю версію. На це є причини:

Різний API. У 1-й і 2-й версіях є невеликі відмінності, які доведеться вивчати. Безліч багів і швидкість виконання. Траплялося так, що старі версії показували більш високі результати швидкості роботи. Але версія 2.3 нарешті обігнала своїх попередників.

Основною відмінністю між версіями є перехід з відкладених обчислень (lazy) на миттєві (Eager). Крім того 2-й версія об'єднана з іншим фреймворків глибокого навчання - Keras.

Суть відкладених обчислення полягає в тому, що все тензорні операції відбуваються тільки в момент компіляції. Це трохи відрізняється від стилю програмування на Python, оскільки результат можна подивитися відразу ж, тому TensorFlow 2, слідом за своїм конкурентом Pytorch, перейшов на режим миттєвих обчислень (Eager mode). Хоча розробники все ж залишили можливість переходу на відкладені обчислення.

Фреймворк глибокого навчання Keras перекочував в TensoFlow 2, і тепер його функціонал зберігається в модулі tf.keras. Даний модуль містить інструменти для побудови моделі, такі як:

– Шари (layers), включаючи повнозв'язну (Dense), сверточних (Conv1D), рекурентний (RNN, LSTM). Всього налічується 105 шарів.

– Функції активації (activations), включаючи Softmax, ReLU, Sigmoid. Деякі ж функції активації, наприклад, PReLU, доступні у вигляді шарів. Всього їх 15.

– Функції втрат (losses), включаючи Binary Crossentropy, MAE, MSE. Всього їх 25.

– Метрики (metrics), наприклад, MAE, MSE, Cosine Similarity. Всього 43 у вигляді класів і 25 у вигляді функцій.

– Оптимізатори (optimizers), включаючи SGD, Adam, Adagrad. Всього їх 9.

– Зворотні виклики (callbacks), включаючи RemoteMonitor для передачі подій на сервер, History для відстеження історії подій моделі, EarlyStopping для зупинки моделі в разі відсутності поліпшень, TensorBoardдля активації режиму візуалізації. Всього їх 14.

2.5. Опис структури програми та алгоритмів її функціонування

Крок 1. Захоплення зображення в якості вхідного з камери

Зображення з веб-камери буде використовуватися як вхідні дані. Отже, щоб отримати доступ до веб-камери, було створено нескінченний цикл, який буде захоплювати кожен кадр. Використовувався метод cv2.VideoCapture (0), що надає OpenCV доступ до веб-камери, після чого використовувався для захоплення зображення метод cap. cap.read (), котрий зчитує кожен кадр і зберігає зображення в змінній кадру.

Крок 2. Виявлення особи на зображенні і створення області інтересу (ROI)

Щоб виявити обличчя на зображенні, потрібно спочатку перетворити зображення у відтінки сірого, оскільки алгоритм OpenCV для виявлення об'єктів приймає сірі зображення на вході. Інформація про колір для виявлення

об'єктів не потрібна. Далі буде використовуватися каскадний класифікатор Хаара для виявлення осіб. Цей рядок використовується для установки нашого класифікатора

- face = cv2.CascadeClassifier (шлях до нашого xml файлу haar cascade').

Потім виконується виявлення, завдяки:

- faces = face.detectMultiScale(gray).

Функція повертає масив виявлення з координатами x, y, висотою і шириною прямокутника об'єкта. Тепер можна перебирати межі й малювати граничні рамки для кожної грані.

- for (x, y, w, h) in faces:
- cv2. rectangle(рамка, (x, y) , (x + w, y + h) , (100 , 100 , 100) , 1)

Крок 3. Виявлення очей в ROI й передання їх класифікатору

Та ж процедура виявлення осіб використовується і для виявлення очей. Спочатку ми встановлюємо каскадний класифікатор для очей в eyes відповідно, потім визначаємо очі, використовуючи eyes = eye.detectMultiScale(gray). Тепер потрібно виділити тільки дані про очі з повного зображення. Це можна зробити шляхом виокремлення граничного прямокутника очей, а потім виділити зображення ока з кадру за допомогою цього коду.

- _eye = frame [y : y + h, x : x + w]

_eye містить тільки дані зображення очі. Це буде завантажено в наш класифікатор CNN, який визначить, відкриті чи закриті очі.

Крок 4. Класифікатор виявляє, чи є очі відкритими або закритими

Для прогнозування стану очей використано класифікатор CNN. Щоб передати зображення в модель, потрібно виконати певні операції, тому що для початку моделі потрібні правильні розміри. Спочатку кольорове зображення конвертується у відтінки сірого, використовуючи:

- _eye = cv2.cvtColor (_eye, cv2.COLOR_BGR2GRAY).

Потім змінюється розмір зображення до 24 * 24 пікселів, оскільки наша модель навчалася на зображеннях 24 * 24 пікселів:

- cv2.resize (_eye, (24,24)).

Для кращої збіжності дані треба нормалізувати:

- _eye = _eye / 255 (всі значення будуть у інтервалі 0-1).

Завантажимо модель, використовуючи:

- model = load_model ('models / cnnCat2.h5').

Тепер ми прогнозуємо кожне око за допомогою нашої моделі:

- epred = model.predict_classes (_eye).

Якщо значення epred [0] = 1, це означає, що очі розплющені, якщо значення epred [0] = 0, то це означає, що очі заплющені.

Крок 5. Підрахуємо бал, щоб визначити сонливість людини

Оцінка – це в основному значення, яке буде використовуватися, щоб визначити, як довго водій заплющує очі. Таким чином, якщо обидва ока закриті, оцінка продовжить збільшуватися, а коли очі відкриті – зменшуватися. Результат виводиться на екран за допомогою функції cv2.putText (), яка буде відображати стан людини в реальному часі:

- cv2. putText (frame, «Open», (10 , 20) , font, 1 , (255 , 255 , 255) , 1 , cv2. LINE_AA)

Граничне значення визначається, наприклад, якщо оцінка стає більше 15, що означає, що очі людини заплющені протягом тривалого періоду часу. Тоді ми подаємо сигнал за допомогою sound.play()

2.6.Обґрунтування та організація вхідних та вихідних даних програми

В якості вхідних даних є дані відеоряду від камери та вхідні дані моделі для розпізнавання обличчя та очей.

В якості вихідних даних є результат аналізу забраження і відповідно да цих результатів - текстове та звукове повідомлення системи (повідомлення про результати наведено на рисунках 2.4 - 2.7).

2.7. Опис розробленого програмного продукту

2.7.1. Використані технічні засоби

Для написання кваліфікаційної роботи та прогорамного забезпечення використовувався ноутбук Ноутбук Dell Vostro 15 3500 знаступними технічними характеристиками: екран 15.6" WVA (1920x1080) Full HD, / Intel Core i3-1115G4 (3.9 — 4.1 ГГц) / RAM 8 ГБ / SSD 256 ГБ / Intel UHD / без ОД / LAN / Wi-Fi / Bluetooth / вебкамера.

2.7.2. Використані програмні засоби

Для написання коду використовувався редактор тексту Sublime Text 3, позаяк він дуже простий і інтуїтивно зрозумілий у використанні, а також підтримує плагіни для мови Python.

Sublime Text - це багатоплатформовий текстовий редактор, розроблений для користувачів, які шукають ефективний, але мінімалістський інструмент для редагування коду. Редактор, звичайно ж, простий, в якому відсутні панелі інструментів або діалогові вікна.

Інструменти Sublime Text. Sublime Text пропонує безліч функцій, які спрощують компіляцію коду.

Goto Anything - це зручна функція, яка дозволяє легше отримувати доступ до файлів. Для переходу до аспектам скомпільованої коду, таким як символи, рядки або слова, потрібно всього кілька дій.

Зіставлення дужок - це функція, яка дозволяє швидко визначити неправильне зіставлення. Редактор напряму виділяє відповідні набори дужок.

Множинне виділення - це зручний інструмент, який дозволяє швидко змінювати рядки коду на ходу. Змінюйте імена змінних або навіть файлів.

Наявність потужного Python API в Sublime виділяє текстовий редактор серед конкурентів. Потужний вбудований API дозволяє Sublime досягати

більшої функціональності, дозволяючи плагинам розширювати вбудовану функціональность.

Кросплатформеність - Sublime Text доступний на кількох клієнтських комп'ютерах, включаючи Windows, Mac і Linux.

Sublime Text - це легкий текстовий редактор, який підійде будь-якому програмісту. Програма зроблена зі швидкістю, що знаходиться в її основі. Особливість програми в її швидкості і чуйності призначеного для користувача інтерфейсу.

У редакторі є безліч плагінів, які інтегруються в одному місці.

Повністю настроюється - текстовий редактор створений, щоб дозволити кінцевому користувачеві легко «пограти» з ПО на свій лад. Sublime дозволяє налаштовувати безліч функцій, включаючи: прив'язки клавіш, меню, фрагменти, макроси і багато інших. Крім того, змінюйте зовнішній вигляд, налаштувавши свої теми для ПО.

Кроссплатформенная підтримка - в редакторі доступна на більшості поширених настільних клієнтів, включаючи Windows, macOS і Linux.

Sublime з відкритим вихідним кодом, відповідно безкоштовний.

З редактором, можна комфортно перемикатися між різними файлами. До того ж, завдяки функції Goto Anything, доступ до якої отримуєте безпосередньо з клавіатури за допомогою клавіш Ctrl або Command + P.

Простота у використанні. Редактор підходить для будь-якого користувача, незалежно від рівня його досвіду.

2.7.3. Виклик та завантаження програми

Запускатися проєкт буде за допомогою Bash-Scripting, сценарію командного рядка, написаного для оболонки Bash. Оскільки Bash – командна оболонка GNU/Linux, для того, щоб запустити скрипт на Windows, потрібен спеціальний інтерфейс командного рядка. Було використано Cygwin, Unix-оточення для Windows. Для запуску програми потрібно створити спеціальний

файл із розширенням .sh, в якому будуть прописані усі потрібні команди. А саме: установка усіх необхідних пакетів для Python та запуск самої програми.

Спочатку треба відкрити командний рядок та перейти до папки із проєктом:

- cd C:\Users\User\Desktop\Detection2

Далі слід виконати команду bash, щоб відкрити середовище Cygwin. Для того, щоб запустити скрипт, треба зробити його виконуваним. Для цього виконуємо команду:

- chmod +x Script.sh

І щоб нарешті запустити скрипт виконуємо команду:

- ./Script.sh

Встановлення пакетів, відкриття веб-камери та початок виявлення може тривати кілька секунд.

Результати роботи наведених команд показано на рис. 2.2–2.3.

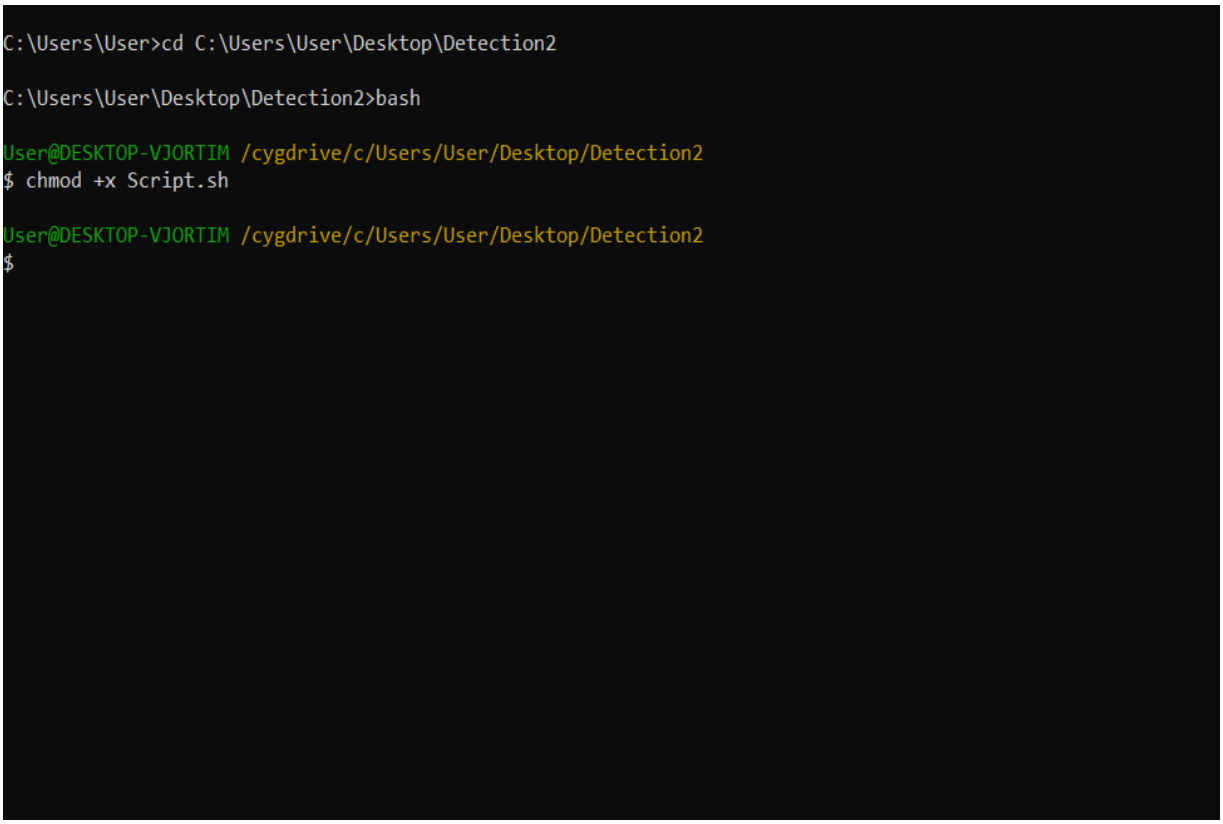

Рис 2.2. Запуск командного рядка, перехід у каталог проєкту, запуск середовища Cygwin і надання скрипту права на виконання

```
User@DESKTOP-VJORTIM /cygdrive/c/Users/User/Desktop/Detection2
$ ./Script.sh
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: opencv-python in c:\program files\python36\lib\site-packages (4.4.0.44)
Requirement already satisfied: numpy>=1.13.3 in c:\program files\python36\lib\site-packages (from opencv-python) (1.18.5
)
WARNING: You are using pip version 20.2.4; however, version 20.3.3 is available.
You should consider upgrading via the 'c:\program files\python36\python.exe -m pip install --upgrade pip' command.
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: tensorflow in c:\program files\python36\lib\site-packages (2.2.0)
Requirement already satisfied: grpcio>=1.8.6 in c:\program files\python36\lib\site-packages (from tensorflow) (1.32.0)
Requirement already satisfied: absl-py>=0.7.0 in c:\program files\python36\lib\site-packages (from tensorflow) (0.10.0)
Requirement already satisfied: numpy<2.0,>=1.16.0 in c:\program files\python36\lib\site-packages (from tensorflow) (1.18
.5)
Requirement already satisfied: keras-preprocessing>=1.1.0 in c:\program files\python36\lib\site-packages (from tensorflo
w) (1.1.2)
Requirement already satisfied: termcolor>=1.1.0 in c:\program files\python36\lib\site-packages (from tensorflow) (1.1.0)

Requirement already satisfied: tensorflow-estimator<2.3.0,>=2.2.0 in c:\program files\python36\lib\site-packages (from t
ensorflow) (2.2.0)
Requirement already satisfied: google-pasta>=0.1.8 in c:\program files\python36\lib\site-packages (from tensorflow) (0.2
.0)
Requirement already satisfied: astunparse==1.6.3 in c:\program files\python36\lib\site-packages (from tensorflow) (1.6.3
)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\program files\python36\lib\site-packages (from tensorflow) (3.3.0
)
Requirement already satisfied: wheel>=0.26; python_version >= "3" in c:\program files\python36\lib\site-packages (from t
ensorflow) (0.35.1)
Requirement already satisfied: wrapt>=1.11.1 in c:\program files\python36\lib\site-packages (from tensorflow) (1.12.1)
```

Рис 2.3. Запуск проєкту

2.7.4. Опис інтерфейсу користувача

Знімки екрану з результатами виявлення заплющених чи відкритих очей показано на рис. 2.4 – 2.7.

В процесі тестування виявлено недостатньо точну роботу системи при виявленні стану очей людини в окулярах в порівнянні з роботою без окулярів, що пояснюється тим, що навчання здійснювалося на основі знімків людей без окулярів.

Усунути проблему можна за допомогою додаткового навчання системи за допомогою іншої бази знімків.

На рисунку 2.4. наведено зображення результата аналізу виявлення розплющених очей в окулярах.

Рис 2.4. Виявлення розплющених очей в окулярах

На рисунку 2.5. наведено зображення результата аналізу виявлення заплющених очей в окулярах.

Рис 2.5. Виявлення заплющених очей в окулярах

На рисунку 2.6. наведено зображення результата аналізу виявлення розплющених очей без окулярів.

Рис 2.6. Виявлення розплющених очей без окулярів

На рисунку 2.7. наведено зображення результата аналізу виявлення заплющених очей без окулярів.

Рис 2.7. Виявлення заплющених очей без окулярів

Оскільки дана розробка планувалась як програма що повинна мати можливість інтеграції в наявну систему транспорту, що має бути простою і надійною у використанні, то додаткові вимоги щодо інтерфейсу та перевантаженність візуальними елементами не ставилися.

Також слід зазначити що програма є невимогливою до програмно-апаратних засобів.

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1. Розрахунок трудомісткості та вартості розробки програмного продукту

1) Передбачувана кількість операторів: 1200;
2) Коефіцієнт складності програми: 1,4;
3) Коефіцієнт корекції програми в ході розробки: 0,08;
4) Годинна заробітна плата програміста, грн./год.: 125 грн./год.;
5) Вартість машино-години, грн./год.: 15 грн./год.;

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_о + t_и + t_а + t_п + t_{отл} + t_д \text{, людино-годин} \quad (3.1)$$

де $t_о$- витрати праці на підготовку й опис поставленої задачі (приймається 50);

$t_и$- витрати праці на дослідження алгоритму рішення задачі;

$t_а$- витрати праці на розробку блок-схеми алгоритму;

$t_п$- витрати праці на програмування по готовій блок-схемі;

$t_{отл}$- витрати праці на налагодження програми на ЕОМ;

$t_д$- витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q * C * (1 + p), \quad (3.2)$$

де q - передбачуване число операторів (1200);

C - коефіцієнт складності програми (1,4);

р - коефіцієнт кореляції програми в ході її розробки (0,07).

Маємо:

$$Q = 1200 * 1,4 * (1 + 0,08) = 1814,4 \approx 1814 \text{ операторів}$$

Витрати праці на вивчення опису задачі $t_и$визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_и = \frac{Q * B}{(75..85) * k}, \text{ людино-годин} \quad (3.3)$$

де В - коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі (1,2);

k - коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності (0,8).

Маємо:

$$t_и = \frac{1814 * 1,2}{75 * 0,8} = \frac{2177,28}{60} = 36,29 \text{ , людино-годин} \quad (3.4)$$

Витрати праці на розробку алгоритму рішення задачі:

$$t_а = \frac{Q}{(20..25) * k}, \text{ людино-годин}$$

Маємо:

$$t_а = \frac{1814}{20 * 0,8} = 113,38 \text{ , людино-годин}$$

Витрати на складання програми по готовій блок-схемі:

$$t_п = \frac{Q}{(20..25) * k}, \text{ людино-годин} \quad (3.5)$$

Маємо:

$$t_п = \frac{1814}{20 * 0,8} = 113,38 \text{ , людино-годин}$$

Витрати праці на налагодження програми на ЕОМ:

$$t_{отл} = \frac{Q}{(4..5) * k}, \text{ людино-годин} \quad (3.6)$$

Маємо:

$$t_{отл} = \frac{1814}{5 * 0,8} = 453,5 \text{ , людино-годин}$$

Витрати праці на підготовку документації:

$$t_д = t_{др} + t_{до}, \text{ людино-годин} \quad (3.7)$$

де $t_{др}$- трудомісткість підготовки матеріалів і рукопису:

$$t_{др} = \frac{Q}{(15..20) * k}, \text{ людино-годин} \quad (3.8)$$

$t_{до}$- трудомісткість редагування, печатки й оформлення документації

$$t_{до} = 0,75 * t_{др}, \text{ людино-годин} \quad (3.9)$$

Маємо:

$$t_{др} = \frac{1814}{15 * 0,8} = 151,16 \text{ , людино-годин}$$

$$t_{до} = 0,75 * 151,16 = 113,375 \text{ , людино-годин}$$

$$t_д = 151,16 + 113,375 = 264,535 \text{ , людино-годин}$$

Маємо наступну трудомісткість розробки ПЗ:

$$t= 50+36,29+113,38+113,38+453,5+264,535 \approx 1\ 031 \text{ , людино-годин,}$$

3.2. Розрахунок на створення програмного забезпечення

Витрати на створення ПЗ $К_{по}$включають витрати на заробітну плату виконавця програми $З_{зп}$і витрат машинного часу, необхідного на налагодження програми на ЕОМ.

$$К_{по} = З_{зп} + З_{мв}, \text{ грн} \quad (3.10)$$

Заробітна плата виконавців визначається за формулою:

$$З_{зп} = t * С_{пр}, \text{ грн} \quad (3.11)$$

де: t- загальна трудомісткість, людино-годин (1 031);

$С_{пр}$- середня годинна заробітна плата програміста, грн/година (125)

Маємо:

$$З_{зп} = 1\ 031 * 125 = 128\ 875, \text{ грн}$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{мв} = t_{отл} * С_{мч}, грн \quad (3.12)$$

де $t_{отл}$- трудомісткість налагодження програми на ЕОМ, год (453,5).

$С_{мч}$- вартість машино-години ЕОМ, грн/год (15).

Маємо:

$З_{мв} = 453,5 * 15 = 6\ 802,5$, грн

Маємо наступні витрати на створення програмного продукту:

$К_{по} = 128\ 875 + 6\ 802,5 = 135\ 677,5$, грн

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення АСУП.

Очікуваний період створення ПЗ:

$$Т = \frac{t}{B_k * F_p}, міс \quad (3.13)$$

де B_k- число виконавців (1);

F_p- місячний фонд робочого часу (при 40 годинному робочому тижні F_p=176 годин).

$$Т = \frac{1031}{1*176} = 5,85 \approx 6, міс$$

Висновок: в кваліфікаційній роботі було розраховано трудомісткість розробки програмного забезпечення, очікуваний період створення програмного забезпечення та витрати на створення програмного забезпечення. Трудомісткість склала 1031 людино-годин, очікуваний період створення склав приблизно 6 місяців, витрати на створення програмного продукту склали 135 677,5 грн., з яких 128 875 грн. – витрати на заробітну плату виконавців, а 6 802,5 грн. – вартість машинного часу.

ВИСНОВКИ

Аналіз сучасних тенденцій розвитку систем виявлення сонливості за кермом показує, що дана проблема є актуальною на сьогоднішній момент, однак такі системи недостатньо поширені через їх важкодоступність для вітчизняних водіїв.

Порівняння наявних на ринку аналогів дає підстави стверджувати, що найпопулярніші автовиробники надають такі системи, але вони не є уніфікованими та сумісними одна з одною.

Враховуючи недоліки розглянутих систем, було сформульовано вимоги до власного проєкту, серед яких можна виділити кросплатформеність та низькі системні вимоги.

Програма виявляє сонливість водія шляхом виділення заплющених очей на зображені, одержаному з камери. В процесі розробки було використано OpenCV для виявлення очей за допомогою каскадного класифікатора Хаара, а також модель CNN для прогнозування стану.

Під час тестування виявилося, що використаний алгоритм працює недостатньо точно, якщо водій носить окуляри. Для вирішення цієї проблеми планується додаткове навчання системи з використанням знімків людей в окулярах.

Актуальність інформаційної системи визначається великим попитом на подібні розробки.

Всі посталені в роботі задачі було виконано в повному обсязі

Також у кваліфікаційній роботі було визначено трудомісткість розробленого програмного продукту (1031 люд-год), проведений підрахунок вартості роботи по створенню програми (135 677,5 грн.) та розраховано час на його створення (6 міс.).

Впровадження даного програмного продукту є економічно вигідним, оскільки його повна вартість є помірною за рахунок використання

некомерційних інструментів для розробки, відсутності необхідності конфігурації і обслуговування робочих місць користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Медичні заходи для попередження ДТП, які сталися внаслідок підвищеної сонливості водія [Електронний ресурс]. – Режим доступу: https://core.ac.uk/reader/279341090

2. Епідеміологічне дослідження хропіння да ОСАГС серед 374 водіїв вантажівок у Гуанчжоу, Китай [Електронний ресурс]. – Режим доступу: https://pubmed.ncbi.nlm.nih.gov/23803540/

3. Робота з OpenCV [Електронний ресурс]. – Режим доступу: https://docs.opencv.org/master/d9/df8/tutorial_root.html

4. Робота з TensorFlow [Електронний ресурс]. – Режим доступу: https://www.tensorflow.org/tutorials

5. Робота з Keras [Електронний ресурс]. – Режим доступу: https://keras.io/guides/

6. Робота з Pygame [Електронний ресурс]. – Режим доступу: https://www.pygame.org/docs/

7. Xml файли [Електронний ресурс]. – Режим доступу: https://github.com/opencv/opencv/tree/master/data/haarcascades

8. CNN файл [Електронний ресурс]. – Режим доступу: https://github.com/LvivHacker2/CNN

9. Виведення зображення на екран [Електронний ресурс]. – Режим доступу: https://www.youtube.com/watch?v=JwIaqOYbe7c

10. Виявлення обличчя та очей на зображенні [Електронний ресурс]. – Режим доступу: https://www.youtube.com/watch?v=88HdqNDQsEk

11. Fatigue Detection Warning [Електронний ресурс]. – Режим доступу: https://gb.e-guide.renault.com/eng/Koleos-2/FATIGUE-DETECTION-WARNING

12. Driver Attention Warning [Електронний ресурс]. - Режимдоступу: http://www.miamilakesautomall.com/kia-blog/why-you-should-get-the-kia-drive-wise-package/

13. IBuzz Alert [Електронний ресурс]. – Режим доступу: https://drivetribe.com/p/skoda-safety-ibuzz-fatigue-alert-f43lX1BbTT2Mn-XhfC26jg?iid=P-yO7ZqcSJu_H25lgYfyaw

14. Cygwin. Bash-Scripting [Електронний ресурс]. – Режим доступу: https://www.youtube.com/playlist?list=PL7KBbsb4oaOnz1FOlCDof2oSgEE-YDywp

15. https://4pda.ru/forum/index.php?showtopic=256741 [Електронний ресурс]. – Режим доступу:ADAS-систем є рішення iOnRoad3

16. http://masters.donntu.org/2019/fknt/kozhukhov/diss/indexu.htm [Електронний ресурс]. – Режим доступу: моніторинг і контроль втоми водія

17. Бизли Д. Python. Подробный справочник / Д. Бизли. – СПб. : Символ-Плюс, 2010. – 864 с.

18. Хахаев И. А. Python. Практикум по алгоритмизации и программированию на Python / И. А. Хахаев. – М. : Альт Линукс, 2010. – 126 с.

19. Любанович Б. Python. Простой Python. Современный стиль программирования / Б. Любанович. – СПб. : Питер, 2016. – 480 с.

20. Федоров Д. Ю. Основы программирования на примере языка Python : учеб.пособие / Д. Ю. Федоров. – СПб. : Питер, 2016. – 176 с.

21. Лутц М. Изучаем Python, 4-е издание / М. Лутц. – СПб. : Символ-Плюс, 2011. – 1280 с.

22. Доусон М. Программируем на Python / М. Доусон. – СПб. : Питер, 2014. – 416 с.

23. Мусин Д. Самоучитель Python. Выпуск 0.2 / Д. Мусин. – Pythonworld.ru, 2015. – 136 с.

ДОДАТОК А

ЛІСТИНГ ПРОГРАМИ

```
import cv2
import os
from keras.models import load_model
import numpy as np
from pygame import mixer
import time


mixer.init()
sound = mixer.Sound('alarm.wav')

face = cv2.CascadeClassifier('haar cascade files\haarcascade_frontalface_alt.xml')
leye = cv2.CascadeClassifier('haar cascade files\haarcascade_lefteye_2splits.xml')
reye = cv2.CascadeClassifier('haar cascade files\haarcascade_righteye_2splits.xml')


lbl=['Close','Open']

model = load_model('models/cnncat2.h5')
path = os.getcwd()
cap = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_COMPLEX_SMALL
count=0
score=0
thicc=2
rpred=[99]
lpred=[99]

while(True):
    ret, frame = cap.read()
    height,width = frame.shape[:2]

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
    faces =
face.detectMultiScale(gray,minNeighbors=5,scaleFactor=1.1,minSize=(25,25))
    left_eye = leye.detectMultiScale(gray)
    right_eye =  reye.detectMultiScale(gray)

    cv2.rectangle(frame, (0,height-50) , (200,height) , (0,0,0) , thickness=cv2.FILLED
)

    for (x,y,w,h) in faces:
        cv2.rectangle(frame, (x,y) , (x+w,y+h) , (100,100,100) , 1 )

    for (x,y,w,h) in right_eye:
        r_eye=frame[y:y+h,x:x+w]
        count=count+1
        r_eye = cv2.cvtColor(r_eye,cv2.COLOR_BGR2GRAY)
        r_eye = cv2.resize(r_eye,(24,24))
        r_eye= r_eye/255
        r_eye=  r_eye.reshape(24,24,-1)
        r_eye = np.expand_dims(r_eye,axis=0)
        rpred = model.predict_classes(r_eye)
        if(rpred[0]==1):
            lbl='Open'
        if(rpred[0]==0):
            lbl='Closed'
        break

    for (x,y,w,h) in left_eye:
        l_eye=frame[y:y+h,x:x+w]
        count=count+1
        l_eye = cv2.cvtColor(l_eye,cv2.COLOR_BGR2GRAY)
        l_eye = cv2.resize(l_eye,(24,24))
        l_eye= l_eye/255
        l_eye=l_eye.reshape(24,24,-1)
        l_eye = np.expand_dims(l_eye,axis=0)
        lpred = model.predict_classes(l_eye)
        if(lpred[0]==1):
            lbl='Open'
        if(lpred[0]==0):
            lbl='Closed'
```

```
        break

    if(rpred[0]==0 and lpred[0]==0):
        score=score+1
        cv2.putText(frame,"CLOSED",(10,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
    # if(rpred[0]==1 or lpred[0]==1):
    else:
        score=score-1
        cv2.putText(frame,"OPEN",(10,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)


    if(score<0):
        score=0
    cv2.putText(frame,'Score:'+str(score),(100,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
    if(score>15):
        #person is feeling sleepy so we beep the alarm
        cv2.imwrite(os.path.join(path,'image.jpg'),frame)
        try:
            sound.play()

        except:  # isplaying = False
            pass
        if(thicc<16):
            thicc= thicc+2
        else:
            thicc=thicc-2
            if(thicc<2):
                thicc=2
        cv2.rectangle(frame,(0,0),(width,height),(0,0,255),thicc)
    cv2.imshow('frame',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

В роботі використано 3 файла моделей:

- haarcascade_frontalface_alt
- haarcascade_lefteye_2splits
- haarcascade_righteye_2splits

<opencv_storage>
<cascade type_id="opencv-cascade-classifier"><stageType>BOOST</stageType>
<featureType>HAAR</featureType>
<height>20</height>
<width>20</width>
<stageParams>
<maxWeakCount>213</maxWeakCount></stageParams>
<featureParams>
<maxCatCount>0</maxCatCount></featureParams>
<stageNum>22</stageNum>
<stages>
<_>
<maxWeakCount>3</maxWeakCount>
<stageThreshold>8.2268941402435303e-01</stageThreshold>
<weakClassifiers>
<_>
<internalNodes>
0 -1 0 4.0141958743333817e-03</internalNodes>
<leafValues>
3.3794190734624863e-02 8.3781069517135620e-01</leafValues></_>
<_>
<internalNodes>
0 -1 1 1.5151339583098888e-02</internalNodes>
<leafValues>
1.5141320228576660e-01 7.4888122081756592e-01</leafValues></_>
<_>
<internalNodes>
0 -1 2 4.2109931819140911e-03</internalNodes>
<leafValues>
9.0049281716346741e-02 6.3748198747634888e-
01</leafValues></_></weakClassifiers></_>
<_>
<maxWeakCount>16</maxWeakCount>

<stageThreshold>6.9566087722778320e+00</stageThreshold>
<weakClassifiers>
<_>
<internalNodes>
0 -1 3 1.6227109590545297e-03</internalNodes>
<leafValues>
6.9308586418628693e-02 7.1109461784362793e-01</leafValues></_>
<_>
<internalNodes>
0 -1 4 2.2906649392098188e-03</internalNodes>
<leafValues>
1.7958030104637146e-01 6.6686922311782837e-01</leafValues></_>
<_>
<internalNodes>
0 -1 5 5.0025708042085171e-03</internalNodes>
<leafValues>
1.6936729848384857e-01 6.5540069341659546e-01</leafValues></_>
<_>
<internalNodes>
0 -1 6 7.9659894108772278e-03</internalNodes>
<leafValues>
5.8663320541381836e-01 9.1414518654346466e-02</leafValues></_>
<_>
<internalNodes>
0 -1 7 -3.5227010957896709e-03</internalNodes>
<leafValues>
1.4131669700145721e-01 6.0318958759307861e-01</leafValues></_>
<_>
<internalNodes>
0 -1 8 3.6667689681053162e-02</internalNodes>
<leafValues>
3.6756721138954163e-01 7.9203182458877563e-01</leafValues></_>
<_>
<internalNodes>
0 -1 9 9.3361474573612213e-03</internalNodes>
<leafValues>
6.1613857746124268e-01 2.0885099470615387e-01</leafValues></_>
<_>
<internalNodes>

0 -1 10 8.6961314082145691e-03</internalNodes>
<leafValues>
2.8362309932708740e-01 6.3602739572525024e-01</leafValues></_>
<_>
<internalNodes>
0 -1 11 1.1488880263641477e-03</internalNodes>
<leafValues>
2.2235809266567230e-01 5.8007007837295532e-01</leafValues></_>
<_>
<internalNodes>
0 -1 12 -2.1484689787030220e-03</internalNodes>
<leafValues>
2.4064640700817108e-01 5.7870548963546753e-01</leafValues></_>
<_>
<internalNodes>
0 -1 13 2.1219060290604830e-03</internalNodes>
<leafValues>
5.5596548318862915e-01 1.3622370362281799e-01</leafValues></_>
<_>
<internalNodes>
0 -1 14 -9.3949146568775177e-02</internalNodes>
<leafValues>
8.5027372837066650e-01 4.7177401185035706e-01</leafValues></_>
<_>
<internalNodes>
0 -1 15 1.3777789426967502e-03</internalNodes>
<leafValues>
5.9936738014221191e-01 2.8345298767089844e-01</leafValues></_>
<_>
<internalNodes>
0 -1 16 7.3063157498836517e-02</internalNodes>
<leafValues>
4.3418860435485840e-01 7.0600342750549316e-01</leafValues></_>
<_>
<internalNodes>
0 -1 17 3.6767389974556863e-04</internalNodes>
<leafValues>
3.0278879404067993e-01 6.0515749454498291e-01</leafValues></_>
<_>

```
      <internalNodes>
        0 -1 18 -6.0479710809886456e-03</internalNodes>
      <leafValues>
        1.7984339594841003e-01 5.6752568483352661e-
01</leafValues></_></weakClassifiers></_>
    <_>
      <maxWeakCount>21</maxWeakCount>
      <stageThreshold>9.4985427856445312e+00</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 19 -1.6510689631104469e-02</internalNodes>
          <leafValues>
            6.6442251205444336e-01 1.4248579740524292e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 20 2.7052499353885651e-03</internalNodes>
          <leafValues>
            6.3253521919250488e-01 1.2884770333766937e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 21 2.8069869149476290e-03</internalNodes>
          <leafValues>
            1.2402880191802979e-01 6.1931931972503662e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 22 -1.5402400167658925e-03</internalNodes>
          <leafValues>
            1.4321430027484894e-01 5.6700158119201660e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 23 -5.6386279175058007e-04</internalNodes>
          <leafValues>
            1.6574330627918243e-01 5.9052079916000366e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 24 1.9253729842603207e-03</internalNodes>
          <leafValues>
            2.6955071091651917e-01 5.7388240098953247e-01</leafValues></_>
```

<_>
<internalNodes>
0 -1 25 -5.0214841030538082e-03</internalNodes>
<leafValues>
1.8935389816761017e-01 5.7827740907669067e-01</leafValues></_>
<_>
<internalNodes>
0 -1 26 2.6365420781075954e-03</internalNodes>
<leafValues>
2.3093290627002716e-01 5.6954258680343628e-01</leafValues></_>
<_>
<internalNodes>
0 -1 27 -1.5127769438549876e-03</internalNodes>
<leafValues>
2.7596020698547363e-01 5.9566420316696167e-01</leafValues></_>
<_>
<internalNodes>
0 -1 28 -1.0157439857721329e-02</internalNodes>
<leafValues>
1.7325380444526672e-01 5.5220472812652588e-01</leafValues></_>
<_>
<internalNodes>
0 -1 29 -1.1953660286962986e-02</internalNodes>
<leafValues>
1.3394099473953247e-01 5.5590140819549561e-01</leafValues></_>
<_>
<internalNodes>
0 -1 30 4.8859491944313049e-03</internalNodes>
<leafValues>
3.6287039518356323e-01 6.1888492107391357e-01</leafValues></_>
<_>
<internalNodes>
0 -1 31 -8.0132916569709778e-02</internalNodes>
<leafValues>
9.1211050748825073e-02 5.4759448766708374e-01</leafValues></_>
<_>
<internalNodes>
0 -1 32 1.0643280111253262e-03</internalNodes>
<leafValues>

3.7151429057121277e-01 5.7113999128341675e-01</leafValues></_>
<_>
<internalNodes>
0 -1 33 -1.3419450260698795e-03</internalNodes>
<leafValues>
5.9533137083053589e-01 3.3180978894233704e-01</leafValues></_>
<_>
<internalNodes>
0 -1 34 -5.4601140320301056e-02</internalNodes>
<leafValues>
1.8440659344196320e-01 5.6028461456298828e-01</leafValues></_>
<_>
<internalNodes>
0 -1 35 2.9071690514683723e-03</internalNodes>
<leafValues>
3.5942441225051880e-01 6.1317151784896851e-01</leafValues></_>
<_>
<internalNodes>
0 -1 36 7.4718717951327562e-04</internalNodes>
<leafValues>
5.9943532943725586e-01 3.4595629572868347e-01</leafValues></_>
<_>
<internalNodes>
0 -1 37 4.3013808317482471e-03</internalNodes>
<leafValues>
4.1726520657539368e-01 6.9908452033996582e-01</leafValues></_>
<_>
<internalNodes>
0 -1 38 4.5017572119832039e-03</internalNodes>
<leafValues>
4.5097151398658752e-01 7.8014570474624634e-01</leafValues></_>
<_>
<internalNodes>
0 -1 39 2.4138500913977623e-02</internalNodes>
<leafValues>
5.4382127523422241e-01 1.3198269903659821e-01</leafValues></_></weakClassifiers></_>
<_>

. . . .

<_>
<rects>
<_>
7 9 6 9 -1.</_>
<_>
7 12 6 3 3.</_></rects></_>
<_>
<rects>
<_>
0 14 2 3 -1.</_>
<_>
0 15 2 1 3.</_></rects></_>
<_>
<rects>
<_>
11 12 1 2 -1.</_>
<_>
11 13 1 1 2.</_></rects></_>
<_>
<rects>
<_>
4 3 8 3 -1.</_>
<_>
8 3 4 3 2.</_></rects></_>
<_>
<rects>
<_>
0 4 20 6 -1.</_>
<_>
0 4 10 6 2.</_></rects></_>
<_>
<rects>
<_>
9 14 1 3 -1.</_>
<_>
9 15 1 1 3.</_></rects></_>
<_>

<rects>
<_>
8 14 4 3 -1.</_>
<_>
8 15 4 1 3.</_></rects></_>
<_>
<rects>
<_>
0 15 14 4 -1.</_>
<_>
0 17 14 2 2.</_></rects></_>
<_>
<rects>
<_>
1 14 18 6 -1.</_>
<_>
1 17 18 3 2.</_></rects></_>
<_>
<rects>
<_>
0 0 10 6 -1.</_>
<_>
0 0 5 3 2.</_>
<_>
5 3 5 3 2.</_></rects></_></features></cascade>
</opencv_storage>

ДОДАТОК Б

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

ДОДАТОК В

ПЕРЕЛІК ФАЙЛВ НА ДИСКУ

Ім’я файла	Опис
Пояснювальні документи	
Пояснювальна_записка.doc	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
Пояснювальна_записка.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF.
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
Презентація.pptx	Презентація кваліфікаційної роботи.