МІНІСТЕРСТВО ОСВІТИ І НАУКИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ВИПУСКНА РОБОТА

на тему:

«Система детекції та розпізнавання обличь»

Завідувач
Випускаючої кафедри **Довбиш А.С.**

Керівник роботи **Шелєхов І.В.**

Студента групи Інз-61с **Оніпка А.П.**

СУМИ 2020

РЕФЕРАТ

Записка: 62 стор, 1 додаток, 15 джерел.

Об’єкт дослідження — детекція та розпізнавання обличь.

Мета роботи — розробити інформаційну систему для детекції та розпізнавання обличь.

Методи дослідження — метод детекції за каскадами Хаара та розпізнавання за власними поверхнями облич.

Результати — створено інформаційне забезпечення системи.

Зміст

ВСТУП

У сучасному світі різноманітні системи для розпізнавання облич призначені з метою автоматичної ідентифікації особистості за допомогою відеопотоків. Такі ситеми виробляють не лише розпізнавання рис облич та образів,з метою перевірки завчасно створених фотоілюстрацій у базах даних. Насамперед винайдені системи для подібних цілей обробляють інформацію за допомогою відеокамери і функції детектору рис облич. Спершу системна функція детектування перевіряє на співпадіння при появі в кадрі персони і фіксує зображення тієї чи іншої особи. Зазвичай можливі декілька варіацій алгоритмів для роботи конкретного модуля розпізнавання,але в основному приміняються фіксація та розпізнавання. В інформаційних ланка дана технологія не нова,але досить розповсюджена та використовувана на даний момент.

Є методи, які використовані в системах розпізнавання осіб, при цьому вони базуються на порівнянні і особистих рис обличчя певної персони конкретного зображенням,і інших наявних факторів ,з обличчями та вцілому фото збереженими в базах даних. Такий підхід описується біометричне доповнення базуючись на основі дій інтелекту, котрий у свою чергу має здатність виключно розпізнати особу за допомогою аналізу моделей на основі текстур обличчя та його форми. На даний момент подібні ІС(інтелектуальні системи) можуть бути використаними навіть в робототехніці для координації,не кажучи про несчисленні мобільні додатки,хоча ще нещодавно були застосовуваними лиша як технологічні додатки. У сучасності набуває популярність застосування таких технологій для контролю та перевірки доступу в конкретних напрямах як:відбитки пальців ,чітке вирізнення окової оболонки, та інші біометричнимі функціональності електронної безпеки. Зазвичай похибка точності подібних технологій розпізнавання облич та емоцій як технології є більшою, на відміну від визначення відокремленості відбитків або зображення ока, не без допомоги безконтактного процессу використання технологій є більш точними для цих цілей. Безперечно також те що подібні технології на-

бувають шаленої популярності не тільки для комерційної діяльності,а також для засобу маркетингу. Різноманітні додатки включають в себе елементи взаємодії персони з персноальним комп'ютером, засобами відеоспостереження, автоматичними методами індексування певних зображень.

В даній роботі розглядається можливість створення інтелектуальної системи ідентифікації особи за допомогою графічних бібліотек OpenCV. У моєму випадку бібліотека в достатній мірі надає засоби щодо обробки і аналізу саме вмісту зображень, включаючи розпізнавання та детекцію силуетів на фотографіях таких як особи і фігури людей,будівль,детелай, тексту тощо, відстежування напряму руху об'єктів, перетворення зображень, пристосування моделей поведінки машинного вдосконалення і виявлення тотожних або різних деталей на різних або подібних фотографіях. Використання технології фіксування та визначення обличча буде продемонстровано надалі у програмному забезпеченні даного дипломного проекту.

1 АНАЛІЗ ПРОБЛЕМИ. ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Сучасний стан і тенденція розвитку технології розпізнавання

Для того щоб сформулювати задачу,потрібно точно розуміти дефініцію поняття - «Система розпізнавання облич». Будь-яка система розпізнавання обличчя - технологія, яка має здатність перевіряти або ідентифіковувати персону з цифрового зображення або відеокадру з джерела відео. Є декілька методів, в яких відтворені системи розпізнавання обличчя, але в цілому вони працюють, порівнюючи вибрані риси обличчя із заданого зображення з обличчями у динамічних базах даних. Вони описуються як програмні технології на основі біометричного штучного інтелекту дозволяючи з певною вірогідністю зафіксувати з розрізненням людей кроками аналізу шаблонів, заснованих на фактурі та формі обличчя людини.

Незважаючи на те, що спочатку це форма комп'ютерних застосувань, останнім часом спостерігається ширше використання на мобільних платформах та в інших формах технології(робототехніка або при застосуванні штучного інтелекту для самоуправління). Але найпоширенішим способом використання є контроль доступу в системах безпеки і він може порівнюватися з іншими біометричними показниками.

У 1997 р. Крістофом фон дер Мальсбургом та аспіранти Університету Бохума Німеччини і Каліфорнії в США розробили інтелектуальну систему,яка перевершувала більшість подібних систем із Массачусетського технологічного інституту та Мерілендського університету. Вона була розроблена за рахунок фінансування дослідницької лабораторії збройних сил США. ПЗ(програмне забезпечення)продавалося як ZN-Face і використовувалося клієнтами, такими як Deutsche Bank та іншими операторами аеродромі,а також іншими комерційними або державними підприємствами. Програмне забезпечення було в досталь надійним для того, щоб робити ідентифікацію з менш ідеального вигляду обличчя. Це також часто можна побачити через такі перешкоди для ідентифікації такі як: вуса, бороди, змінена зачіска та окуляри і навіть сонцезахисні окуляри". При випробуваннях систем викорис-

товувались образи облич дуже високої роздільної здатності, тривимірні сканування та зображення райдужки. Дослідження стверджували про те що сучасні системні алгоритми в 10 разів точніші, аніж застарілі системи визначення обличча 2002 року та в 100 разів точніші за показники минулого десятиліття. Деякі алгоритми мали здатність перевершити людських учасників у визначенні персон, а також однозначно ідентифікувати близнюків або людей зі схожими рисами.

Для ідентифікації осіб були виокремлені декілька перешкод, основними були дві конкретні проблеми. Спершу треба розуміти, що сам піксель певної території картинки в деякому сенсі нічого не означає (зміна кольору майжу не гарантує різниці для всього зображення,не кажучи про помітність такої заміни для людського ока). Звичайно, проблема кількість піке слів при,особливо при високів розмірності картинки, один на фоні інших,має мАле значення. Інша складність полягає у тому що будь-яка картинка являє собою масив пікселів. З чого можна зробити висновок що конкретне встановлення зображень являє собою надлишковість та неекономічність. Щодо ефективності визначення персон, то тут слід опрацювати саме певний компактний і зручний спосіб подачі фотозображення. На сьогодення у світі відомо несчисленні засоби для утискання файлів які мають відношення до ілюстрованих зображень з такими форматами як (pdf,png,jpg,bmp та ін.) із певними згубленними даними,які не впливають на цілісність файлів, але проблема використаного формата полягає у не зручності при класифікації особ на фото, з причин таких як вирішення задачі відокремлення одних осіб від інших, для яких потрібно набагато менше інформації. Пов'язано це в першу чергу з тією умовою, при котрій відсутня необхідность визначати, як виглядає ця персона з вибірки, а потрібно розв’язати інше завдання яке полягае у питанні – яка особа виглядає певним чином. Інша перешкода скаладається так, що одна й та сама особа може бути зафіксованою при різних зовнішніх чинниках та деталях,як приклад можна навести світло у кімнаті або на вулиці,задній фон,позицінування обє’ктів,вираз міміки і навіть шуми зафіксовані камерою.

Переважно складність поставленної задачі полягає у розпізнаванні людей по зображенню обличчя ,яка діляться на деякі складові: пошук у масивних базах, контроль доступу та безпака фотографій в документах. Використання даних складових при захваті та розпізнаванні особистостей вже авно використовується сучасними онлайн сервісами. Такі складові розрізняються як за вимогами, що надаються до систем розпізнавання, так і щодо способів вирішення. У сучасний час налічується незчисленна кількість завершених інформаційних продуктів, маючи багате напрямлення вони посідають місце використання і у визначенних рис обличь. Більша кількість з відомих є комерційними продуктами і інформація про алгоритмічне забезпечення цих систем не набуває публічного доступу. Найвідоміші з них – VOCORD FaceControl 3D і DeepFace. Дані ПЗ існують як комерційні розробки і деталі цих інтелектуальних систем(ІС) не доступні ані у вигляді програмного коду, ані у вигляді формальних документацій. З чого і виникала склідність даного проекту, оскільки переважна кількість інформації носить або поверхневий характер, або загальну схематичну відсутність роботи алгоритмів або їх зразків. Тому як загальний опис,можна зауважити що розробка таких систем займає багато людських і фінансових ресурсів, тому можливість кардинально змінити рушій сили у цих напрямах мають змогу лише великі корпорації,наукові відділи, не згадуючи про те, що якість ткого програмного забезпечення буде переважати у компаній з ресурсами або з державною підтримкою у сфері безпеки, на відміну від локальних онлайн серсів. Але навіть у подібних додатках детектція та розпізнавання у системі VOCORD FaceControl 3D є ключовими принципами. Такі ключові елементи засноване на синхронних знімках стереокамерами з різних ракурсів, побудові 3D-моделі обличчя і порівняні 3Dмоделі з 3D-моделлю у базі або з звичайною фотографією. Цей принцип і не обійшов наслідування цього дипломного проекту. Зрозумівши як працюють примітивні алгоритми таких програм можна наступним чином. Як приклад можна допускати, що фотозображення мають розмірмірність 100 * 100 пікселів з 256 відтінками сірого. Існує безліч варіацій визначення задач.

Варіантом є один з подібних зразків наведених нижче. Вважаємо, є певна кількість зразків для тренування(а саме підвищення вірогідності розрізнення того чи іншого обличчя від не обличчя або від іншого обличчя). Нехай,зразки налічуються до 400 фотографій (по 10 фотографій для 40 людей за різних умов). В такому випадку задачу виокремлення є сенс виокреслити слідуючим образом. Наявне нове зображення в обумовленому вище форматі. Потрібно отримати відповідь з декількох наведених варіацій далі. Як зразок можна привести що дане фотозображення не являє собою обличча людини. Але це не є гарантією, що воно являє собою обличчям конкретної персони з массиву. Далі алгоритм може виявити риси облич,які не містяться в колекції. Маючи уявлення про майбутнє програмне забезпечення цього дипломного проекту, на власному прикладі, то можна сказати що рішення такого завдання було б додання зображення до колекції. Сама ж робота алгоритму досить проста. Спочатку механізм алгоритму відокремлює риси лиця особи з фону. Далі фотозображення нормалізується. Механізм містить слідуючі кроки:

- Зміна дозволу зображення до 100 * 100 пікселів
- Заміна кольорів до 256 відтінків сірого
- Перегляд комплексної насиченості картинки до певних усереднених значень.

Деякі алгоритми потребують вертикалізації зображень. У випадку зміни кута обличча на потрібний відбувається при нормалізації. Далі алгоритм відокремлює характеристики облич. Дані наративні особливості змістовно залежні від самої функції відокремлення конкретної особи від іншої, оскільки їх приклади будуть зазначені згодом. Оскільки для виокремлення особи наявність картинки є несуттєва, то останнім етапом ідентифікації застосувується класифікатор, який при наявних даних видає відповідь на поставлені вище запитання.

1.2 Методології розпізнавання обличь

Метод Віоли-Джонса та ознаки Хаара

Метод, запропонований Віолою і Джонсом в 2001 році, став справжнім проривом в цій області. З'явившись якраз у той час коли комп'ютерне бачення лише починало свій шлях, цей метод набув великої популярності, і достатньо було потреб у високій точності і серйозної теоретичної основі. Основні принципи, на яких базується метод використання зображень інтегральної уяви, з використанням якої можна зручно знаходити потрібні деталі на картинках,що не займає багато часу, мають значуще місце використання познаки за Хааром, які здійснюють знаходження певних об'єктностей у конкретному районі персони,її риси,емоції в конкретний момент,підставлення застосоване при відборі підходящих ознак для об'єкта який потрібно знайти на певній частці фотографії, класифікатор приймає риси,а також робить висновок щодо наявності або відсутності ознак для швидкого відкидання вікон, де не знайдено обличчя. Найбільш популярний подібний метод для пошуку об'єктів на відео-потоці. Звичайно,існує мала ймовірність помилки при знаходження певних рис. Але навіть при таких умовах ІС добре виконує свою функцію і впізнає особу з урахуванням деякого кута(30 градусі з похибкою).

Зображення яке представлене в інтегральному вигляді - може бути застосованим для знаходження ознак Хаара. Не дивлячись на важливість алгоритмів розпізнавання,потрібно звернути увагу і до фіксації зображень, що стає подальшою основою для роботи таких алгоритмів,в свою чергу Ознаки Хаара являють собою позначенням оцифрованого відбиття, які приміняються у розрізненні образів,силуетів,об'єктів,тощо. Сама ж назва походить від вейвлетів Хаара за візуальною подібністю. Вони були застосованими у найпершому фіксаторі облич,який працював в реальному часі. Ознака Хаара є територією певних дотичних областей у вигляді прямокутників. Ці області мають позиціювання на конкретному зображенні, слідом вираховується сума інтенсивності діючих пікселів,і надалі різниця між сумами. Залишок дій і буде наратимним елементом тієї чи іншої властивості, зазначеної розмірмірності,займаючи позицію на змальованій області. Загалом, для багатьох зображень є позиція, яка в області кожного ока темніша, аніж на позиції щік. З чо-

го можна зробити твердий висновок що загальною ознакою Хаара для лиця являють собою 2 прилеглі прямокутні області, які знаходяться в районах очей і щік.

Порівнюючи ознаку, ключовою особливістю ознак Хаара, є найбільша швидкість. Будучи простою для розуміння обчислюється значення такої ознаки за формулою за такими змінними: $F = X – Y$, де X – сума значень яскравості точок закриваються світлою частиною ознаки, Y – сума значень яскравості точок що закриваються темною частиною ознаки. Ознаки Хаара дають точкове значення перепаду яскравості по осі X і Y відповідно. У даному прикладі функція проводитиме скан прямокутної областіза ознаками Хаара ,що матиме наступний вигляд:

- треба обрати прямокутну область для зчитування та застосувати дані особливості для основних рис обличчя;
- зчитування області зміщується на картинці зі здвигом в 1 клітинку прямокутної області при умові, що розмір самого вікна 24x24 клітинки;
- в кожній області вираховується близ 200 000 варіантів розташування ознак за допомогою зчитування і за допомогою зміни масштабу самих ознак і їх положення у вікні;
- для відмінних маштабувань зчитування відбувається поступово;
- всі знайдені ознаки потрапляють із певною періодичністю у класифікатор, котрий зафіксовує наявність певних рис,наприклад рота, носа, та іншої ознаки певної розмірності у обраній області і виконує пошук найближчого зразка.
- Зразок проходить через перевірку базою даних.

Даний класифікатор мусить відреаговувати виключно на конкретну підмножину всіх ознак, яку він вивчає з особливостей навчального відбору. Системи ієрархічних функцій,інші алгоритми класифікації можуть бути у виді застосованих класифікаторів. Для реагування класифікаторів потрібні вхідні дані такі як: розмір та розташування очей, розмір та розташування рота,

розташування і розмір інших головних рис. З метою покращення роботи відокремлення приміняється технологія *Boosting*. *Boosting* це комплекс методів, які сприяють підвищенню вірогідності розпізнаванняд для точності аналітичних моделей. В його основі закладена побудова каскада класифікаторів, котрі (окрім найпершого) навчаються з урахуванням попередніх помилок інших класифікаторів. Як приклад,можна взяти один з ранніх алгоритмів,який застосовував вибірку із 3-х моделей, перша з яких вдосконалювалась на повному наборі даних, друга – на відбору деяких прикладів,в решті-решт в 50% з яких перша дала вірні варіанти відповідей, а третя – на прикладах, у яких дані найперших двох розминулись.

Метод головних компонентів та алгоритм eigenfaces (власних облич). Нижчепродемонстрований метод має достатньо зручне композиціювання у сучасних програмах, з використанням мінімальних ресурсів. Метод головних компонент (principal component analysis) – застосовується для витягу з потрібної інформації з даних великої розмірності. Графічна ілюстрація(рис. 1.1)

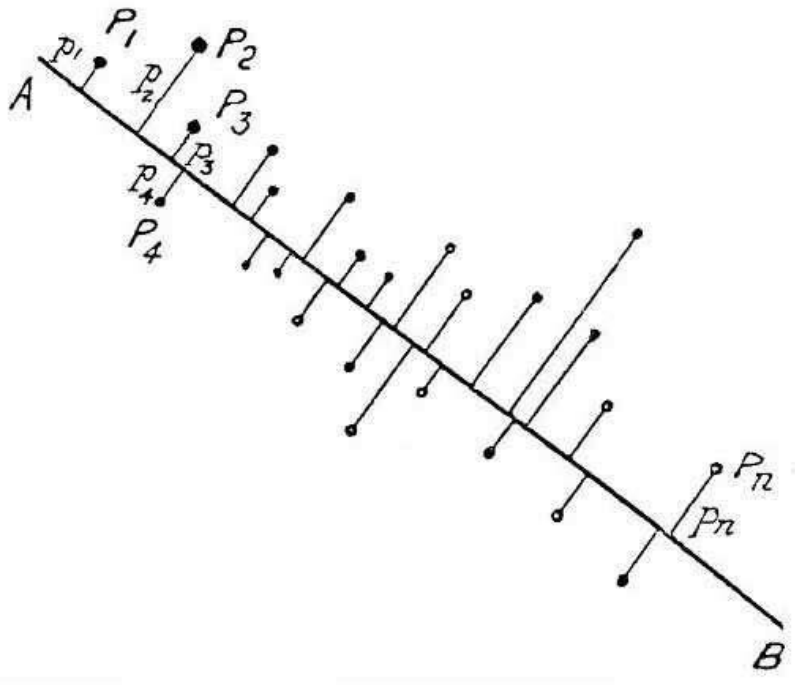


Рисунок 1.1. – вибір компонентів

Мета лежить у застосуванні методу основних компонентів задля значного зменшення розмірності даних. Загалом у багатовимірному випадку процес вибору основних компонентів (рис. 1.1) приблизно такий:

• відпочатку відшукується епіцентр збереження даних, і там передається нове координатне походження - це нульовий головний компонент (PC0);

• вибрано напрямок максимальної зміни даних - це перший основний компонент (PC1);

• якщо вхідна інформація описана не повністю (високий рівень шуму), тоді вибір впадає на інший напрям (PC2) - перпендикулярно першому для опису змін у решті даних,тощо. Рисунок 1.1 – Розходження вхідної інформації на фото, яку сприймає машинне бачення. Ціль таких методів – це істотне зниження розмірності даних. Оскільки велике навантаження буде впливати не тільки на центральний процесор,що буде впливати на якість комп'ютерного бачення, а ще на ефективність роботи алгоритму при величезному вмісті баз даних. Загалом, в багатовимірному випадку, процес виділення головних компонентів (рис. 1.1) відбувається приблизно таким чином:

• шукається центр хмари даних, і там передається нове координатне походження - це нульовий(PC0);

• вибрано напрямок максимальної зміни даних - це перший (PC1);

• якщо дані не описані повністю (високий рівень шуму), вибирається інший напрямок (PC2) - перпендикулярно першому для опису змін у решті даних тощо.

В результаті ми переходимо до нового вигляду, розмір якого значно менший. Часто можна спростити розмір лише до двох. Розпізнаючи обличчя, метод PCA дозволяє відображати великий одновимірний вектор пікселів, побудований з двовимірного зображення одного обличчя в різних версіях, у компактних основних компонентах функціонального простору. Результатом перетворення можна назвати проекційний підпростір. Він обчислюється точ-

но шляхом визначення власних векторів коваріаційної матриці, отриманих із набору зображень об'єктів.и великий одновимірний вектор пікселів, побудований з двовимірного зображення одного обличчя у різних варіантах, в компактні основні компоненти простору ознак. Результат перетворення можна назвати підпростором проекції. Даний додатковий простір розраховується саме шляхом визначення власних векторів коваріаційної матриці, отриманої з набору зображень об’єктів.

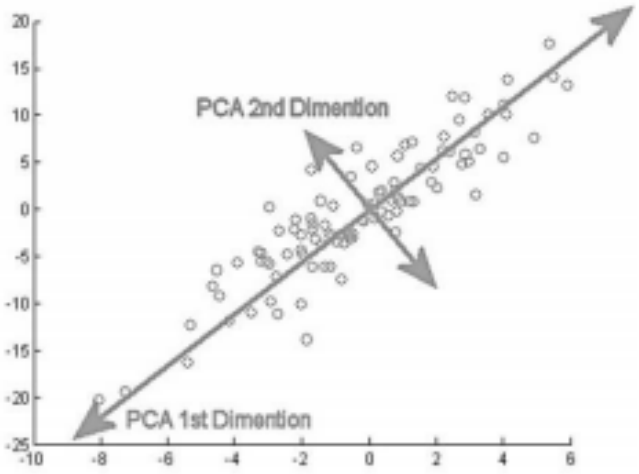


Рисунок 1.2 – Вибір головних компонентів

Метод прихованої моделі Маркова

Прихована Марківська модель здійснює імітацію функціонування процесів, подібних до процесів Маркова із прихованими параметрами. На базі нової моделі встановлюється завдання пошуку цих параметрів за спостережуванними параметрами. Отримані параметри можуть бути використані в подальшому аналізі для розпізнавання обличчя. З точки зору розпізнавання - зображення є двовимірним дискретним сигналом. Важливу роль у побудові моделі зображення відіграє вектор спостереження, тобто обхід зображення. Задля уникнення різниці в дописах, зазвичай використовується вікно прямокутного розрізнення. Вікна прямокутної форми повинні перекриватися один з одним, щоб не втратити області даних. Значення для перекриття, а також об-

ласті розпізнавання підбираються експериментально. Після вилучення пристрою він перетворюється в цифровий блок одним із двох способів. За допомогою методу Карунена-Лоєва (KLT) та дискретного косинусу без конверсії (DCT).

Розпізнавання обличчя на власній поверхні

KLT у своїй базовій формі не дає дуже хороших результатів, хоча модифікації методів, які потребують значних вирахувальних міцностей, не мають подібних недоліків. Наприклад параметр DCT має просту форму і більш стійкий до шуму, перетворення, спотворення. Після отримання блоку пік сіл перекладається у вектор F, який містить лише значущі елементи. Після цього збережені вектори розподіляються відповідно до станів моделі. Державна модель представляє певні класи об'єктів. Всього це 5 наборів, відповідальні областям обличь - спереду, очам, носу, ротові, підборіддя. Процес наступного стану наступає одразу після попереднього, до наступного - після завершення попереднього. Вважається, що обличчя зберігаються в базі даних на різних рівнях абстракції. На найнижчому рівні знаходиться набір пікселів. Цей рівень призначений лише для відштовхування, оскільки його використання в обчислювальній техніці вимагає дуже великих обчислювальних ресурсів. Саме тут вводиться поняття підпису - числовий дескриптор, який описує візуальні характеристики певної області. Переваги використання підписів очевидні, вони потребують значно менших витрат на оновлення конфігурацій комп’ютера. Пошук заснований на порівнянні наборів підписів. Пошук проводиться до тих пір, поки ступінь подібності наборів підписів не відповідає заданим вимогам точності. Практично, тобто реалізація програмного забезпечення базується на трикутному дереві - дійсно фіксованому дереві запитів - структурі для пошуку збігів залежно від критеріїв пошуку. Компоненти цього дерева - це міра відстані, набір ключових зображень та набір елементів бази даних. Краї від кореня до листа визначають індекс листя, листя дерев в свою чергу містять елементи бази даних. Шлях від коре-

ня до аркуша - це відстань від елемента бази даних до кожної з клавіш. За допомогою PCA ви можете повернути кожне оригінальне зображення навчального набору на його поверхні. Важливою особливістю PCA є те, що ви можете комбінувати власні інтерфейси для відновлення будь-якого оригінального зображення з навчального набору. Пам’ятайте, що ваші власні поверхні не менше, ніж риси обличчя. Тому можна сказати, що оригінальне зображення обличчя можна реконструювати з його власних поверхонь, якщо додати всі власні елементи (функції) у правильних пропорціях. Кожна поверхня відображає лише певні риси обличчя, які можуть бути або не бути в оригінальному зображенні. Якщо функція присутня в оригінальному зображенні більшою мірою, частка відповідного власного поверху у «сумі» власних поверхонь повинна бути більшою. Якщо, навпаки, особливість відсутня (або майже відсутня) у вихідному зображенні, то відповідна власна поверхня повинна бути меншою (або зовсім не) частиною до суми її власних поверхонь. Тому для реконструкції вихідного зображення з його власних поверхонь потрібно побудувати зважену суму всіх його власних поверхонь. Тобто реконструйоване оригінальне зображення дорівнює сумі всіх його власних поверхонь, і кожна власна поверхня має певну вагу. Ця вага визначає ступінь, в якій певна особливість (власна поверхня) присутня в оригінальному зображенні. Як це стосується розпізнавання обличчя? Ідея полягає в тому, що ви можете витягнути обличчя з власних поверхонь із заданим набором ваг, але піти іншим шляхом. Протилежний спосіб - видалити лусочки з власних поверхонь та обличчя, які слід розпізнати. Ці луски відрізняються не меншою мірою, оскільки розмір, при якому оглядають обличчя, відрізняється від «типових» граней, представлених власними поверхнями. Тому за допомогою цієї ваги можна зафіксувати:

Визначивши чи справжній образ справді обличчя. Якщо вага зображення занадто відрізняється від ваги зображень обличчя (тобто зображень, з яких ми точно знаємо, що таке обличчя), зображення, ймовірно, не є обличчям.

Такі грані (зображення) мають схожі риси (власні поверхні) з однаковими ступенями (вагами). Якщо витягуєте ваги з усіх доступних зображень, ви можете згрупувати зображення в кластери. Тобто всі зображення, що мають однакову вагу, можуть мати подібні грані. Приклад показаний на малюнку 1.4. Цей метод полягає у побудові тривимірного графіка обличчя, а потім у вигляді сітки на обличчі. Цей метод дуже обчислювальний і лінійно залежить від кількості людей у базі даних, оскільки для навчання нових людей потрібно тривалий час. Приклад рис. 1.4.

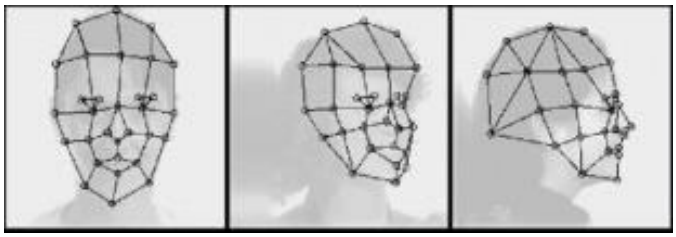


Рисунок 1.4 - метод гнучкого порівняння на графах.

Метод застосування нейронних мереж. Приклад показаний на рис. 1.4. Алгоритм розпізнавання навчається на заздалегідь підготовленій вибірці осіб, проводитися настройка ваг міжнейронних зв'язків в процесі вирішення оптимізаційної задачі методом градієнтного спуску. Даний метод показує гарні результати розпізнавання, але має високу складність і вимагає повного перенавчання при додаванні нового обличчя в базу даних.

Застосування нейромереж

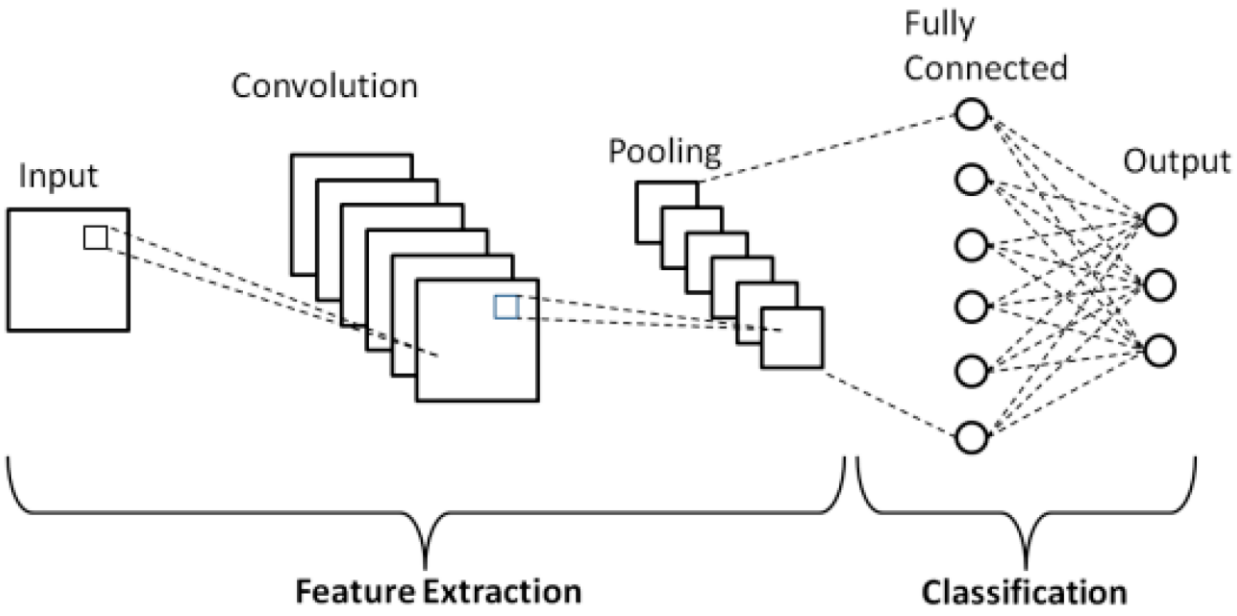


Рисунок 1.5 - метод застосування нейронних мереж.

Завдання розпізнавання обличчя - розрізнити вхідні сигнали (дані зображення) на кілька класів (людина). Вхідні сигнали надзвичайно галасливі (наприклад, шум, спричинений різними умовами освітлення, поставою тощо), але вхідні зображення не є абсолютно випадковими, і, незважаючи на їх відмінності, в будь-якому вхідному сигналі є закономірності. Такими зразками, які можна спостерігати за всіма сигналами, може бути - у сфері розпізнавання обличчя - наявність певних предметів (очей, носа, рота) на будь-якому обличчі, а також відносна відстань між цими об’єктами. Ці характеристики називаються внутрішніми поверхнями в області розпізнавання обличчя (або основними компонентами загалом). Їх можна дістати з вихідного зображення за допомогою математичного інструменту, який називається Principal Analysis (PCA).

1.3 Постановка задачі

Метою роботи є розробка та програмна реалізація системи розпізнавання облич та збереження інформації. Для досягнення поставленої мети необхідно виконати такі завдання:

1) Формування вхідного математичного опису системи детектування та розпізнавання обличь

2) Створення математичної моделі навчання інтелектуальної системи детектуванню обличь на фото.

3) Розробка алгоритму детектування обличь

4) Створення математичної моделі навчання інтелектуальної системи ідентифікації особи за зображенням обличчя.

5) Розробка алгоритму ідентифікації особи

6) Програмна реалізація інформаційної інтелектуальної системи, що здатна виконувати детектування обличь на фото, ідентифікацію особи за ними та збереження одержаних результатів.

7) Тестування розробленої системи

2 ВИБІР МЕТОДА РОЗВ'ЯЗАННЯ ЗАДАЧІ

2.1 Математична модель та алгоритми процесу детектування за ознаками Хаара

Вирізнення кольору обличчя було помилково виявлено через фони з кольоровими компонентами біля кольору обличчя. Орім того, екземпляри дозволяють більшості алгоритмів потрапляти у зазначені проблеми. Запропонована гібридна система розпізнавання обличчя застосовує попередній фільтр для відбору пікселів кольору шкіри, а потім попередньо відфільтроване зображення використовує технологію покриття, аналогічну запропонованій. Можна зробити висновок, що використання подвійних методів виявлення обличчя збільшує час виконання, а потім переходить до автономних алгоритмів. Експерименти в цій роботі показують, що використання цієї гібридної системи змушує її обличчя працювати належним чином і з меншою кількістю пікселів, що означає меншу складність обчислень. Даний алгоритм витягує відповідну інформацію зображення і кодує її максимально ефективну для зчитання різниці між поточним та попереднім результатом. Математично алгоритм обчислює власні вектори матриці коваріації безлічі зображень обличчя. Власні значення квадратичної матриці - прості константи, які представляють всю матрицю в рівнянні власного вектора:

$$Ax=\lambda x \qquad (1)$$

Формула 1 – рівняння власного вектору.

де x - власний вектор, а λ - власне значення. Для кожного власного вектора має місце лише одне асоційоване власне значення. Для обчислення власних векторів квадратної матриці слід розпочати з обчислення власних значень таким чином:

$$Ax- \lambda x = 0$$
$$(A- \lambda l)x=0 \qquad (2)$$
$$|A- \lambda l)=0$$

Формула 2 – рівняння обчислення власних значень

Для початку обчислимо власні значення з рівняння (2), потім будемо використовувати рівняння (1) для отримання відповідних власних векторів. Власні вектори, що відповідають ненульовим власним значенням матриці коваріації, дають ортонормальну основу для зображення підпростору. Власні вектори сортуються у порядку зменшення відповідно до відповідних власних значень. Зовнішній вектор, пов'язаний з найбільшим власним значенням, зазвичай відображає максимальну дисперсію у зображенні кадру відео. З іншого боку, одиниця відносно найменшого власного значення відповідає мінімальній дисперсії. Кожне зображення в наборі вносить свій власний вектор, цей вектор характеризує відмінності між зображеннями. Коли ми представляємо ці власні вектори, ми називаємо їх власними поверхнями. Кожне обличчя можна представити у вигляді лінійної комбінації його поверхонь; однак ми можемо зменшити кількість власних поверхонь до тих, які важливіші, щоб ми могли зробити її більш ефективною. Основна ідея алгоритму - розробити систему, яка може порівнювати не самі кадри, а ці функції. Завдання виявлення обличчя означає пошук усіх прямокутників, що містять усі грані з вихідного матеріалу, з мінімумом додаткових елементів. Успіх існуючих алгоритмів залежить від багатьох факторів, серед яких: постава, кут обличчя, кут камери, освітлення, аксесуари та ілюзії. Таким чином, одним з найпопулярніших алгоритмів є алгоритм Віола-Джонса та його модифікації на основі обчислення функцій Хаара. Цей алгоритм дає надзвичайно хороші результати для лобових граней, тобто для тих, для яких кути повороту [−15 °; 15 °] в горизонтальній площині і [−35 °; 10 °) вертикально. Точність моделей, заснованих на цьому методі, погіршується зі збільшенням кута повороту граней. У цій статті ми дізнаємося емоції лобових облич за допомогою моделей,

заснованих на методі Віоли-Джонса. Таким чином, ми розуміємо, що точність цих моделей залежить від кута повороту, кута камери. Другий аспект - наявність таких аксесуарів, як окуляри, борода, вуса, макіяж тощо, які також допускають помилки в роботі алгоритмів і тому вимагають використання декількох моделей або більш складних моделей. Алгоритми ідентифікації людей, які шукають людські очі, адже кожне обличчя має очі. Але навіть подібний алгоритм можна порівняти з протиставленням, в якому модель може видаті погані результати. Як приклад, обличчя сонцезахисні окулярами. Останній згаданий аспект - це можливість ілюзій. Артефакти які схожі на обличчя людини, але, зокрема, не манекени чи маски. Всі перераховані вище аспекти є надзвичайно важливими для вирішення проблеми розпізнавання обличчя, але це не завдання цієї роботи. Достатньо опрацювати зображення фасадів, без аксесуарів та ілюзій. Як метод виявлення я використовую метод Віоли-Джонса, який був опублікований у 2001 році Полом Віолою та Майклом Джонсом. Ця робота мала суттєвий вплив на розробку алгоритмів розпізнавання зображень, оскільки цей алгоритм, незважаючи на його простоту, першим вирішив проблему розпізнавання облич у режимі реального часу. Ідея цього методу ґрунтувалася на Хаар-подібних особливостях. Знак Хаар - це значення, обчислене за формулою 3.

$$T = \left(\sum_{j=1}^{N} w_j \sum_{(x;y)\in A(j)} I(x,y) \right) / s^2, \qquad (3)$$

Формула 3 - ознаки Хаара

де T – значення ознаки; wj - це константи, що обернено пропорційні до розміру відповідної області (кількості пікселів); N – кількість областей; $I(x, y)$ – це значення пікселя зображення; $A(j)$ – j-та область зображення; s – коефіцієнт маштабування $(s \geq 1)$. Ознаки Хаара визначаються як прямокутники з чорно-білими областями, а значення ознаки, як сума значень пікселів

у чорні області мінус сума пікселів у білих областях. Додатково значення суми з області множиться на ваговий коефіцієнт. Приклад (рис. 2.1)

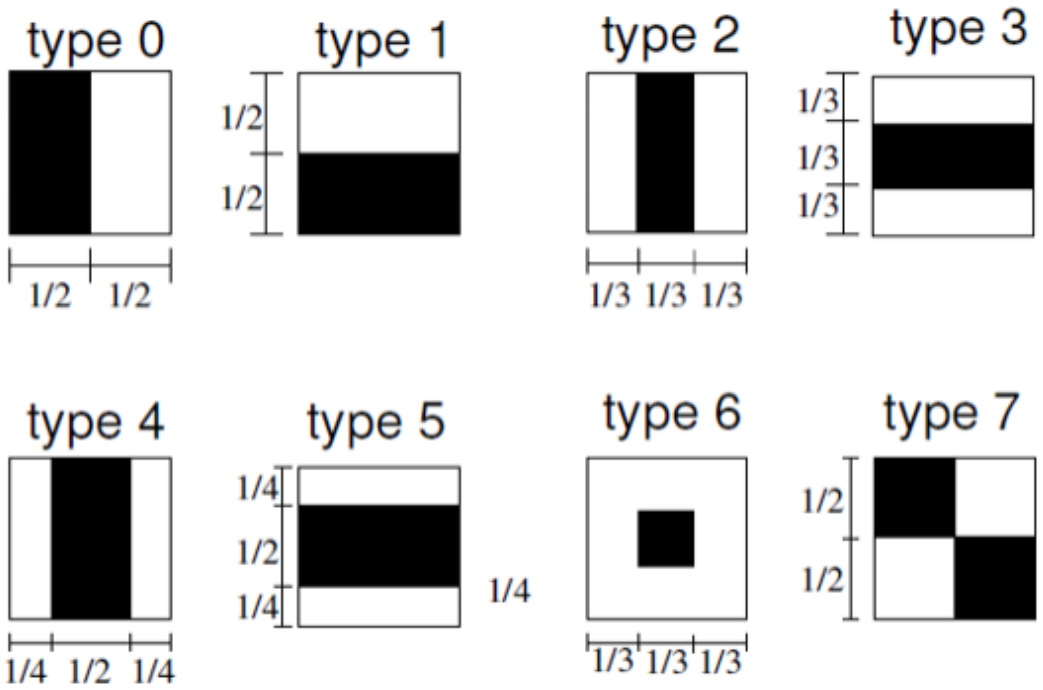


Рисунок 2.1 - Типи ознак Хаара

Такі згортки підкреслюють структурну інформацію об'єкта: наприклад для центру особи людини буде завжди негативна наступна згортка (рис. 2.2)

Рисунок 2.2 - підкреслення негативної згортки на фото

Очі будуть темнішими, ніж область між ними, так само як область рота буде темнішою від обличчя. Чим більше використовуються різні примітиви, тим точніше можлива класифікація об’єкта. Однак якщо точна класифікація не

має такої потреби, то є можливість у використанні значно меншої кількості примітивів. Розрізняючи об’єкт на зображенні в порівнянні із тестовим зображенням. Перевага Хаара, і чому не потрібно використовувати криві, як приклад - синусова хвиля та Гаус, в замін цих прямокутних фігур? Окрім того, каскади Хаара мають дуже швидкий перегляд завдяки інтегрованій візуалізації зображень. Інтегроване подання зображення - це матриця, яка відповідає розміру вихідного зображення. Всякий елемент зберігає суму інтенсивності всіх точок зліва та над цим елементом. Елементи матриці обчислюються за формулою (4):

$$L(x, y) = \sum_{i=0,\, j=0}^{i \le x,\, j \le y} ¿ I(i, j) \qquad (4)$$

Формула 4 – розрахунок елементів матриці

де *I (i, j)* - яскравість пікселя вихідного зображення.Кожен елемент матриці L *(x, y)* є сумою пікселів в прямокутнику від *(0,0)* до *(x, y)*, тобто значення кожного пікселя *(x, y)* дорівнює сумі значень усіх пікселів лівіше і вище даного пікселя *(x, y)*. Розрахунок матриці займає лінійний час, пропорційне числу пікселів в зображенні, тому інтегральне зображення прораховується за один прохід. Розрахунок матриці можливий за формулою (5):

$$L(x,y) = I(x,y) – L(x-1,y-1) + L(x,y-1) + L(x-1,y) \qquad (5)$$

Формула 5 - розрахунок матриці.

За такою інтегральною матриці можна дуже швидко вирахувати суму пікселів довільного прямокутника (рис 2.3), довільної площі. Нехай в прямокутнику *ABCD* є цікавий для нас об'єкт *D*:

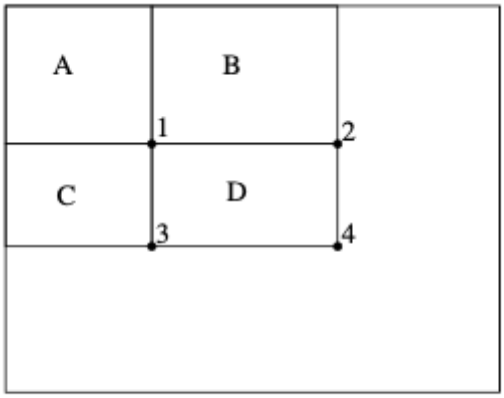


Рисунок 2.3 - приклад прямокутника для вірахування суми пікселів

З малюнка 2.3 зрозуміло, що суму усередині прямокутника можна виразити через суми і різниці суміжних прямокутників за формулою

$$S(ABCD) = L(A) + L(C) — L(B) — L(D) \quad (6)$$

Формула 6 - сума всередені прямокутника

2.2 Математична модель процесу ідентифікації особи за зображенням обличчя за допомогою метода власної поверхні

Система розпізнавання особи складається з двох важливих етапів: функція витягання і класифікація. Особа розпізнавання має завдання виконувати в режимі реального часу. Розпізнавання обличчя може зайняти багато часу, оскільки воно страждає від величезної кількості пікселів. Потрібно зменшити кількість пікселів. Це називається зменшенням розмірності або видаленням функцій, щоб заощадити час для кроку прийняття рішення. Видалення функцій стосується перетворення простору особа в простір функцій. У просторі можливостей база даних осіб представлена зменшеною кількістю функцій, які зберігають більшість важливої інформації оригінального особи. Найпопулярніший метод досягнення - використання алгоритму Eigenface. Алгоритм

Eigenface - це класичний статистичний метод, який використовує лінійне перетворення каруми-Лоєва (KLT), також відомий як аналіз основних компонентів. PCA обчислює власні вектори матриці коваріації вхідного простору межі. Ці власні вектори визначають нове місце людини, де зображені зображення. На відміну від них, N-PCA був розроблений для лінійного PCA. Аналіз основних компонентів аналіз основних компонентів (PCA) [1,2,3,4,5,6,10,11,13,14,15] - це статистичний метод зменшення розмірності, дає оптимальну лінійну розкладання мінімум квадратів навчального набору. У таких програмах, як стиснення зображень і розпізнавання осіб використовується корисна статистична методика під назвою PCA і є спільною методикою визначення шаблонів великих даних. PCA зазвичай називають використанням Eigen face. Потім застосовується підхід PCA для зменшення розмірності даних шляхом стиснення даних і виявлення найбільш ефективних низькомірніх моделей структури особи. Перевага цього зменшення розміру полягає в тому, що він видаляє інформацію, яка не є корисною, і саме розбиває структуру особи на компоненти, які не пов'язані між собою і відомі як особа Ейгена. Кожне зображення особи може зберігатися в 1D-масиві, який представляє зважену суму (вектор функції) граней. У цьому випадку потрібен повний вид спереду особа, інакше результат розпізнавання не буде точним. Основна перевага цього методу полягає в тому, що він може усікати дані, необхідні для визнання підприємства до 1/1000 наявних даних. Після видалення функції наступним кроком є етап класифікації, який використовує евклидову відстань для порівняння придатності тесту і тренованого зображення. На етапі тестування кожне тестове зображення повинно бути середнім центром, тепер спроектуйте тестове зображення в тому ж просторі, яке визначено на етапі навчання. Це прогнозоване зображення зараз порівнюється з проектованим навчальним зображенням у власному просторі. Образи порівнюють із заходами подібності. Навчальний зображення, яке ближче до тестового зображення буде послідовним і знайомим для ідентифікації. Обчисліть віднос-

ну евклидову відстань між тестовим зображенням і реконструйованим зображенням людини, мінімальна відстань дає найкращу відповідність.

Аналіз нормованих основних компонентів (N-PCA)

На відміну від лінійних PCA, N-PCA був розроблений, щоб дати кращі результати в плані ефективності. NPCA - це розширення над лінійним PCA, в якому спочатку нормалізація зображення робиться для видалення варіацій блискавки і фонових ефектів, а однокомпонентні декомпозиції (SVD) використовуються замість внутрішніх значень декомпозицій (EVD) з наступними кроками лінійного вилучення PCA. класифікація. Ваги ступенів розраховуються відповідно до мети. Перш за все, нам потрібно отримати навчальний набір M сірих зображень особи I1, I2, ..., IM. Вони повинні бути: вирівняні особа, очі на одному рівні і межі однаковою шкали, нормалізовані так, що кожен піксель має значення від 0 до 255 (тобто один байт на кодування пікселів) і однаковий розмір N × N., просто виправляючи все формально, ми хочемо отримати безліч {I1, I2, ..., IM}, де

$$I_k = \begin{bmatrix} p^k_{1,1} & p^k_{1,2} & \cdots & p^k_{1,N} \\ p^k_{2,1} & p^k_{2,2} & \cdots & p^k_{2,N} \\ \vdots & & & \\ p^k_{N,1} & p^k_{N,2} & \cdots & p^k_{N,N} \end{bmatrix}_{N \times N} \qquad (7)$$

Формула 7 - Матриця пікселів

Отже, лише фіксуючи все формально, ми хочемо отримати набір

{I1, I2, ..., IM}, д е

т а 0≤pki, j≤255.

Після того, як ми це зробимо, нам слід змінити подання зображення оличчя Ik з матриці $N \times N$ на точку *Гk* у просторі *N2*. Тепер ось, як ми це робимо: ми об'єднуємо всі рядки матриці *Ik* в один великий вектор розмірності *N2*. Чи може це стати простішим? Формальна формула :

$$\Gamma_k = \begin{bmatrix} p^k_{1,1} \\ p^k_{1,2} \\ \vdots \\ p^k_{1,N} \\ p^k_{2,1} \\ p^k_{2,2} \\ \vdots \\ p^k_{2,N} \\ \vdots \\ p^k_{N,1} \\ p^k_{N,2} \\ \vdots \\ p^k_{N,N} \end{bmatrix}_{N\times 1}, \text{where } k = 1, \ldots, M \text{ and } p^k_{i,j} \in I_k \qquad (8)$$

Формула 8 – об’єднання рядків матриці в один вектор розмірності N2. Оскільки нас набагато більше цікавлять характерні риси цих облич, давайте віднімемо все, що спільне між ними, тобто середнє обличчя. Середнє обличчя попередніх середньозважених зображень можна визначити як

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i, \qquad (9)$$

Формула 9 – визначення середнього обличча

Кожна грань відрізняється від середнього за вектором *Φi = Γi − Ψ*. Тепер нам слід спробувати знайти набір ортонормальних векторів, які найкраще описують розподіл наших даних. Необхідними кроками в цьому є:

- Отримати коваріаційну матрицю.

$$C = \frac{1}{M} \sum_{i=1}^{M} \Phi_i \Phi_i^T = AA^T, \text{where } A = [\Phi_1 \Phi_2 \ldots \Phi_M]. \qquad (10)$$

Формула 10 - отримання ковариаційної матриці

- Знайти вектори uk та власні значення λk C.

Однак зауважте тут дві речі: A має розмір *N2 × M*, а отже, матриця C має розмір *N2 × N2*. Щоб поставити речі в перспективу - якщо розмір вашого зображення дорівнює 128 × 128, то розмір матриці C буде 16384 × 16384. Визначення власних векторів та власних значень для матриці такого розміру було б абсолютно нерозв'язним завданням! Простий математичний трюк:

спочатку потрібно обчислити матрицю добутку *L = ATA*, розміром *M × M*. І після цього знайти власні вектори vi, i = 1, ..., M of L (з M-го виміру). При цьому якщо *Lvi = λivi*, то

$$\begin{aligned} ALv_i &= \lambda_i A v_i \Rightarrow \\ AA^T A v_i &= \lambda_i A v_i \Rightarrow \\ CAv_i &= \lambda_i A v_i, \end{aligned} \tag{11}$$

Формула 11 - внутрішня матриця добутку

і, отже, *ui = Avi і λi* - це M-власні вектори (розмір N2) і власне значення C відповідно. Переконавшись у нормалізації до ‖ui‖ = 1. Виявляється, існує спосіб позбутись досить багато власних значень із найменшими власними значеннями, тому залиште лише ті R≤M, які мають найбільші власні значення (тобто, лише ті, що вводять найбільші внесок у дисперсію початкового набору зображень), вони записуються до матриці.

Схематизований приклад роботи алгоритму на рис. 2.4.

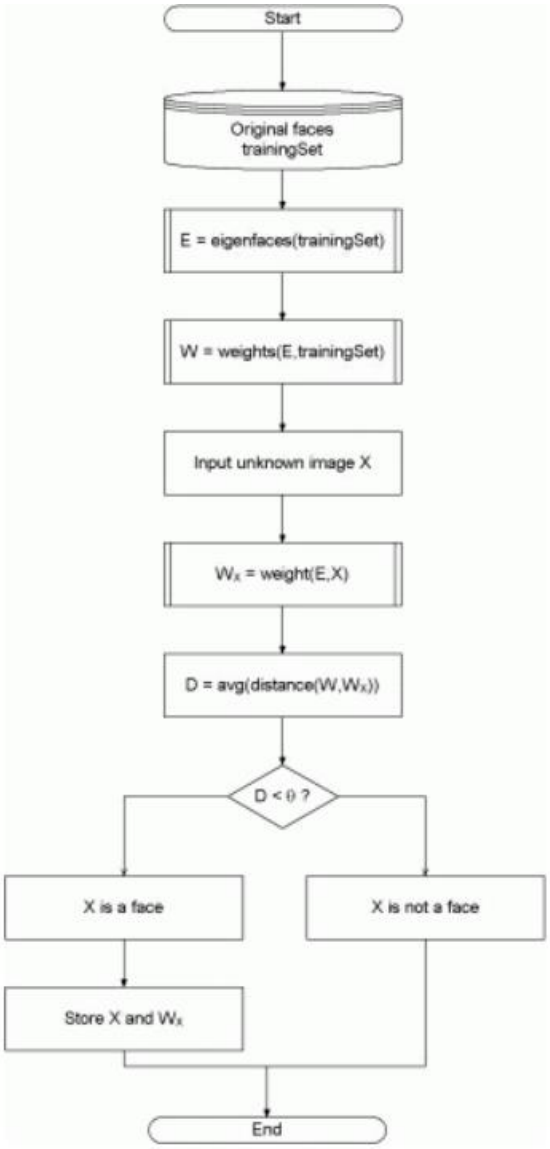


Рисунок 2.4. – схематичне зображення роботи алгоритма.

3 ІНФОРМАЦІЙНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ІНТЕЛЕКТУАЛЬНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Опис вхідних даних

Наразі розрізнення обличь є важливим аспектом проблем безпеки та контролю даних. У цьому проекті розпізнавання обличчя проект поділяється на кілька етапів: виявлення осіб, метод розрізнення та реляційні бази даних. Ідентифікація здійснюється за допомогою Visual Studio 2019, OpenCV та додаткові бібліотеки Emgu. Щоб пов’язати програмне середдовище з бібліотеками OpenCV, необхідно встановити деякі конфігурації ,перш ніж продовжувати програмний процес. Даний проект тримає у собі три основні компонента; Перший - розробка алгоритму виявлення будь-якого обличчя. Другий - це розробка алгоритму, котрий розрізняє обличчя осіб у реальному часі на відео. Останній аспект - аналіз світла замінений на розпізнавання. Коли саме розпізнавання являє собою процес виявлення облич у кадрі та відмежування його від інших об'єктів на задньому плані. Розрізнення проводиться із застосуванням алгоритму Haarcascade. Класифікатор Haarcascade повинене удосконалюватися з кожним попереднім використанням. Метою є отримання високого рівня розпізнавання зображення виявленого обличчя , для цього необхідно обробити у тому місці, де область перетворена з кольорового зображення (RGB), задля зменшення масштабів зображеннь до сірого, потрібно зменшувати дані під час даного процесу. Далі зображення обличчя обрізається та зберігається лише для зменшення фонового шуму. Швидка швидка функція (SURF) використовується для отримання рис обличчя відповідно до зображення для швидшого розрізнення. Для етапу розпізнавання використовується бібліотека Emgu з мовою програмування C#. Під кінець завершення процесу програмування та встановлення обладнання система повинна виявити лицьові сторони та ідентифікувати людей, які також були визначені для того, щоб визначити особу ідентифікованої особи.

3.2 Вибір середовища розробки

Для цього проекту використовувалася остання версія IDE Visual Studio на поточний момент. Дане середовище розробки Visual Studio має зручний функціонал для вдоксконалення, напис знуля, налаштування коду, а також подальшої публікації додатків. Інтегроване середовище розробки (IDE) являє собою багатофункціональну систему , котра може бути використане для різноманітних етапів та аспектів розробки програмного забезпечення. Окрім системного редагування та відладника, які вмонтовані здебільш у саме середовищ IDE, Visual Studio включає в себе вбудовані компіляторні елементи, різноманітні методи допомоги для закінчення коду, інтерфейсні конструкції і

багато інших елементів для спрощення процесу проектування і самої розробки. Критерієм обрання мови програмування C# стала простота реалізації детекції та розпізнавання облич,оскільки дана мова має ефективну співпрацю з бібліотеками OpenCV,що приміняються у цілях даного проекту. Для реалізації программи за допомогою платформи .Net використовувалась технологія Windows Forms.

3.3 Короткий опис програмної реалізації

- Організація робочого місця
- Розробка інтерфейсу користувача
- Підключення необхідних бібліотек та структура даних
- Реалізація алгоритму детекції,розпізнавання та навчання

3.4 Програмна реалізація

Організація робочого місця

Програмний продукт розроблений на комп’ютері з даними характеристиками,що важливо програма не потребує та не використовує значну частину ресурсів даної машини.

Назва параметру	Значення	Мінімальне значення, задане в інструкції системного адміністратора
Процесор	AMD A8	Intel Pentium 233 MHz
ОЗУ	8 GB	128 MB
графічний адаптер	Radeon r200	800×600 (SVGA) with 256 colors
монітор	LOC 2000 1920* 1080	800×600 with 256 colors
операційна система	Windows 10	Windows 2000 SP3 XP or later
Microsoft .NET Framework	4.6.2	3.0
Webcam	Logitech	Будь-яка аналогічна

Розробка інтерфейсу користувача

Форми Windows застосувалися як база основних візуальних елементів даної програми. Додаток з повнофункціональним графічним інтерфейсом, із розширенням та оновленням, дає можливість роботи в наявності або відсутності підключення до Інтернету та використання здебільш безпечнішого доступного ресурсу на локальній системі за порівнянням з іншими додатками Windows. Саме графічне відображення Windows Forms являє собою інтелектуальну клієнтську технологію для .NET Framework, набір певних додаткових бібліотек, використання різноманітних деталей, накшталт використання та запис до файловвї системи. У Windows Forms - це візуальна поверхню, на якій виводиться інформація для користувача. Windows Forms влаштована со-

псобом розміщення та редагування елементів із написання коду для реагування на дії супроводжені натисканням клавіш користувачем. При запиту якої-небудь команди із взаємодією з формою або одним з її компонентів керування впроваджується подія. Редактор додатку на певні дії за допомогою коду котрий дає оброблення під час їх виникненні. Windows Forms включає широкий набір елементів контролю, котрі можна додавати на форми: кнопки, поля, списки, перемикачі та навіть веб-елементи. Набір преважно більшості компонентів керування, котрі можуть бути використані в графічному інтерактиві, знаходяться в таких розділах як елементи контроля які полегшують процес у написанні програмного забезпечення. При умові що цей компонент управління не задовольняє потребам, в Windows Forms можна його відредагувати таким чином створивши власні елементи керування за допомогою класу UserControl. Windows Forms містить багатий функціонал компонентів, які мають місце у застосованні для візуалу майбутньої програми, що надасть перевагу відтворити додатки які мають схожість із Microsoft Office. Застосувуючи елементи керування ToolStrip і MenuStrip, є можливість відтворити меню інструментів,в яке вбудовані і тексти і рнсунки і навіть повноціна панель контролю,не згадуючи багато інших різноманітних елементів керування,які були зазначені вище. Використовуючи контруктор Visual Studio можна легко відображати свої додатки. Достатньо виділити елемент управління за допомогою курсора миші і перемістити його в потрібний район на графічному дисплею форми.

Полегчуючи роботу методом вирішення проблем пов’язаних із вирівнюванням компонентів контролю, дизайнер розробки вміщає інструменти, як смуги вирівнювання клітинки. При застосуванні Visual Studio або компіляції з командної строки можна використовувати різноманітні компоненти при створенні додаткових макетів за досить короткий проміжок часу. Врешті-решт, якщо треба створити компоненти візуальних інтерфейсів, то System.Drawing вміщує широкий спектр класів, необхідних для відтворення

ліній, кіл та інших фігур безпосередньо на формі. Демонстрація інтерфейсу даного програмного продукту програмного показана на рис. 4.1.

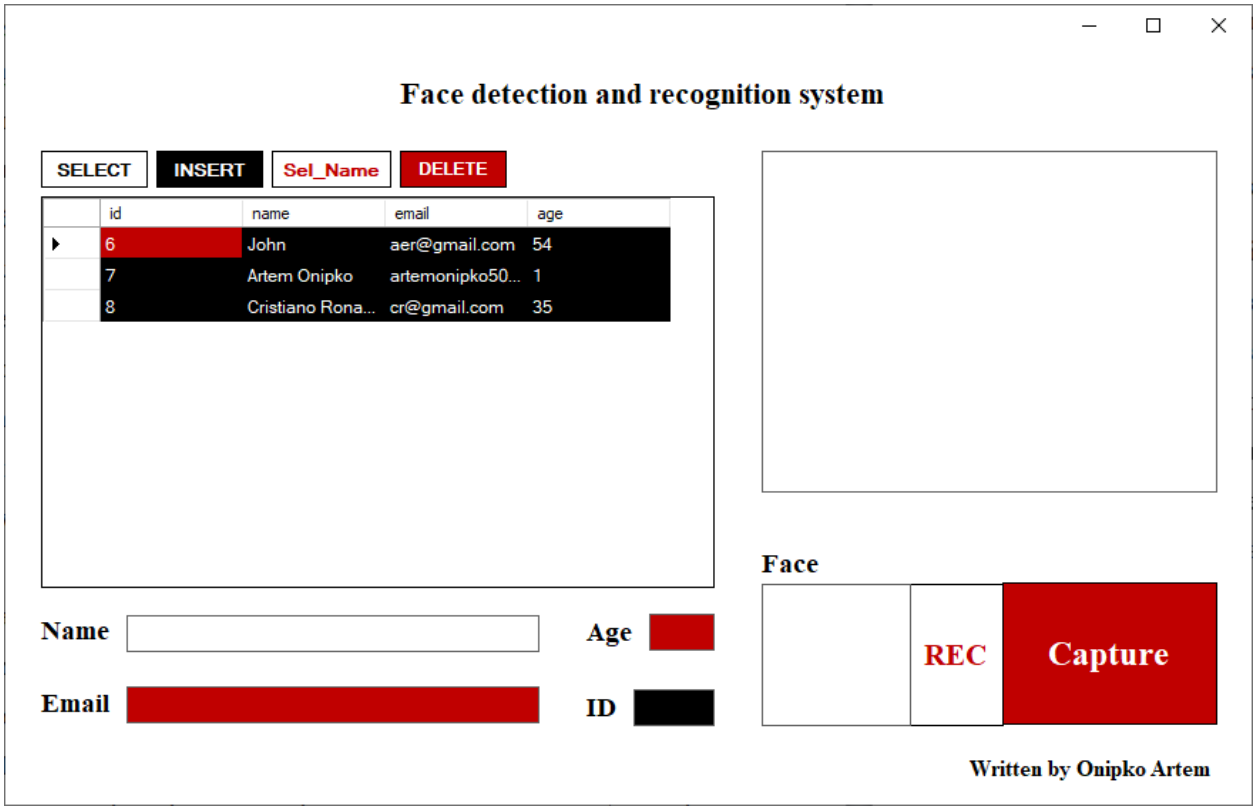


Рисунок 4.1 – інтерфейс головної панелі програми

У правій частині рис 4.2. знаходится інструментарій для детекції та розпізнавання, а також збереження. Кнопка REC необхідна для детекції та розпізнавання які відбуваються у великі області.Кнопка Capture для збереження даних,що також сприяють навчанню алгоритма.Вихідні дані з’являються в малій області. Для наіменування людини потрібно ввести ім’я до текстбоксу. Біля лейблу Detected persons встановлений лічильник котрий обчислює кількість облич на відео.

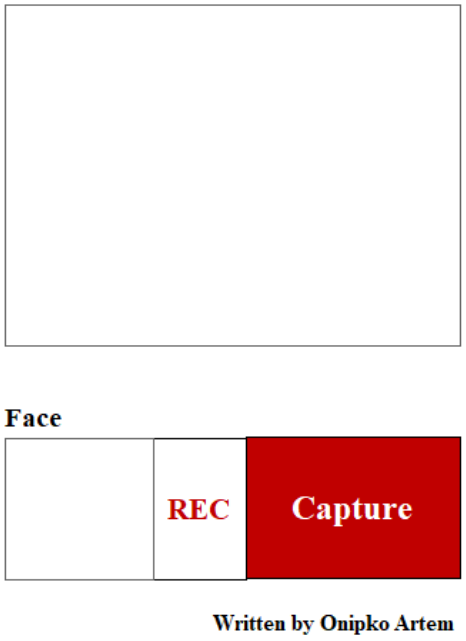


Рисунок 4.2 – частина інтерфейсу відповідальна за детекцію та розпізнавання.

У лівій частині знаходиться інтерфейс для вибірки та редагування бази даних. Велика чорна область datagridview містить базу даних. Для вибірки - SelectDB,додавання – Insert,редагування Update,видалення – Delete. Всі дані будуть відображені у великій області datagridview після заповнення відповідних форм нижче рис 4.3.

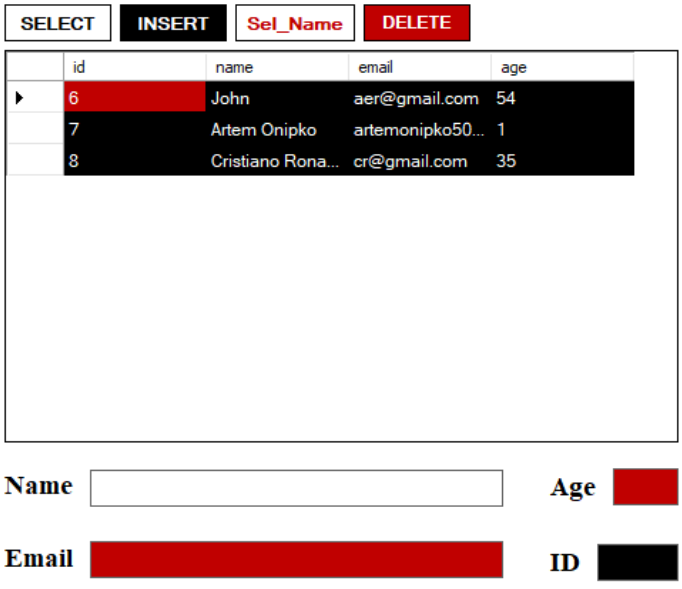


Рисунок 4.3 – частина інтерфейсу відповідальна за редагування бази даних.

Підключення необхідних бібліотек та структура даних

Бібліотеки using System.*

Ббліотека using System; - список імен системи вміщує класи та базові класи, які визначають загальновживані типи значень та довідкові типи даних, події та обробники подій, інтерфейси, атрибути та винятки з обробки.

.Collections.Generic; - список імен *System.Collections.Generic* котрий вміщає в собі різноманітні інтерфейси та класи, які визначають загальні колекції, мають допоміжні чинники у створенні колекцій, які гарантують ліпшу безпеку та ефективність типу, ніж інші колекції сильного типу.

.Drawing; - список *імен System.Drawing* дає доступ до основних функцій графіки *GDI* +. Здебільш розширена функціональність надається в просторах імен *System.Drawing.Drawing2D, System.Drawing.Imaging ma System.Drawing.Text.*

.Windows.Forms; - список імен *System.Windows.Forms* містить класи для створення програм на базі *Windows*, які повністю використовують переваги функцій багатого користувацького інтерфейсу, доступних в операційній системі *Microsoft Windows*.

Using Emgu.CV; - *Emgu CV* - це поперечна платформа .Net обгортка до бібліотеки обробки зображень OpenCV. Дозволяє викликати функції *OpenCV* з *.NET*

System.IO; - список імен *System.IO* містить типи, які дозволяють читати та записувати у файли та потоки даних, а також типи, які забезпечують підтримку базових файлів та каталогів

.Data.SqlClient; - список імен System.*Data.SqlClient* - це постачальник даних *.NET* для *SQL Server*.

.Data; - список імен *System.Data* дає можливість доступу до класів, які представляють архітектуру *ADO.NET*. *ADO.NET* дозволяє створювати елементи, котрі ефективно управляють даними із різноманітних джерел.

Структура бази даних

Microsoft SQL Server - це система управління реляційних баз даних, розроблена Microsoft. Як сервер баз даних, це програмний продукт функцією яка зберігає та отримє дані, на вимогу цього запрошують інші різноманітні програмні програми,це сервер котрий має здатність працювати або на одному тому ж персональному комп'ютері, або на іншому використовуючи мережу (Інтернет включно). В даному проекті база даних використовується для збереження та редагування інформації про людей. Структура бази даних зображена на рис. 4.4.

	Name	Data Type	Allow Nulls
🔑	id	int	☐
	name	nvarchar(50)	☑
	email	nvarchar(50)	☑
	age	tinyint	☑
			☐

Рисунок 4.4. – структура бази даних.

Query - це набір інструкцій, який описує, котру інформацію треба отримати з певного джерела або джерел даних та яку форму та організацію мусять мати поверхневі дані. Запит має різницю від результатів, яку він дає. Зазвичай, вихідні дані систематизовані логічно як послідовність деталей одного типу. Як приклад, таблиця SQL має послідовність рядків. Приклад запиту застосованого в програмі зображений у рис 4.5.

Design | T-SQL

```
1  CREATE TABLE [dbo].[FACES] (
2      [id]    INT           IDENTITY (0, 1) NOT NULL,
3      [name]  NVARCHAR (50) NULL,
4      [email] NVARCHAR (50) NULL,
5      [age]   TINYINT       NULL,
6      PRIMARY KEY CLUSTERED ([id] ASC)
7  );
8
```

Рисунок 4.5. – запит для формування таблицы

Для реалізації даної програми були використані такі ключові слова для запитів як INSERT – команда яка дозволяє додати інформацію до таблиці,Select –

команда яка виконує запит на вибірку значень,та Delete – запит який видаляє значення за первинним ключем. Приклад реалізованої вибірки зображено на рис 4.6.

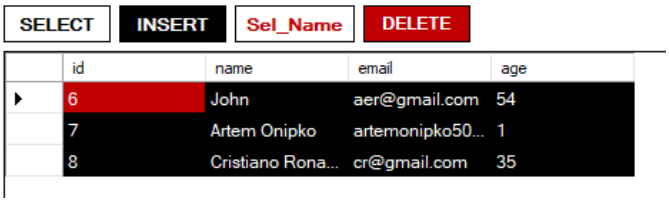


Рисунок 4.6. – вибірка з бази даних.

Реалізація алгоритму детекції та розпізнавання

Задача детекції облич є добре вивченою проблемою у комп'ютерному зорі. Сучасні детектори облич можуть легко знайти та обробити його фронтальні зображення. Нещодавні дослідження у цій області фокусуються на менш контрольованому процесі детекції, де число можливих факторів, такі як зміна пози, вираження емоцій і незвичайне освітлення може призвести до великих візуальних варіацій у вигляді обличчя, що значно погіршує якість роботи детектора облич. Найбільш важкою задачею комп'ютерного зору залишається проблема вирішення неоднозначності, що виникає при перенесенні трьовимірних об'єктів реального простору на плоскі зображення. Дуже багато аспектів залежить від важкопрогнозованих фактів. Складність у детекції облич в основному диктується двома аспектами:

- велика кількість візуальних змін облич людини на
- Різноманітному фоні
- великий розмір простору пошуку можливих позицій облич та їх

Розмірів перший потребує детектору облич точно вирішувати проблему класифікації, коли другий нав'язує проблеми с потребами у швидкодії. Механізм детекції облич на зображеннях знаходить використання у різноманітних додатках, а задача є однією з основоположних у комп'ютерному зорі. Область використання систем розпізнавання осіб, вже не обмежується верифікацією особи і спостереженням. Все більше додатків використовують розпізнавання

осіб як перший крок до інтерпретації дій людини, його намірів і поведінки. Інакше кажучи, до реалізації тих можливостей, які будуть грати центральну роль в інтелектуальному середовищі наступного покоління.В моєму випадку система розпізнавання декількох людей працює після навчання та детекції кожного з них рис.4.7.

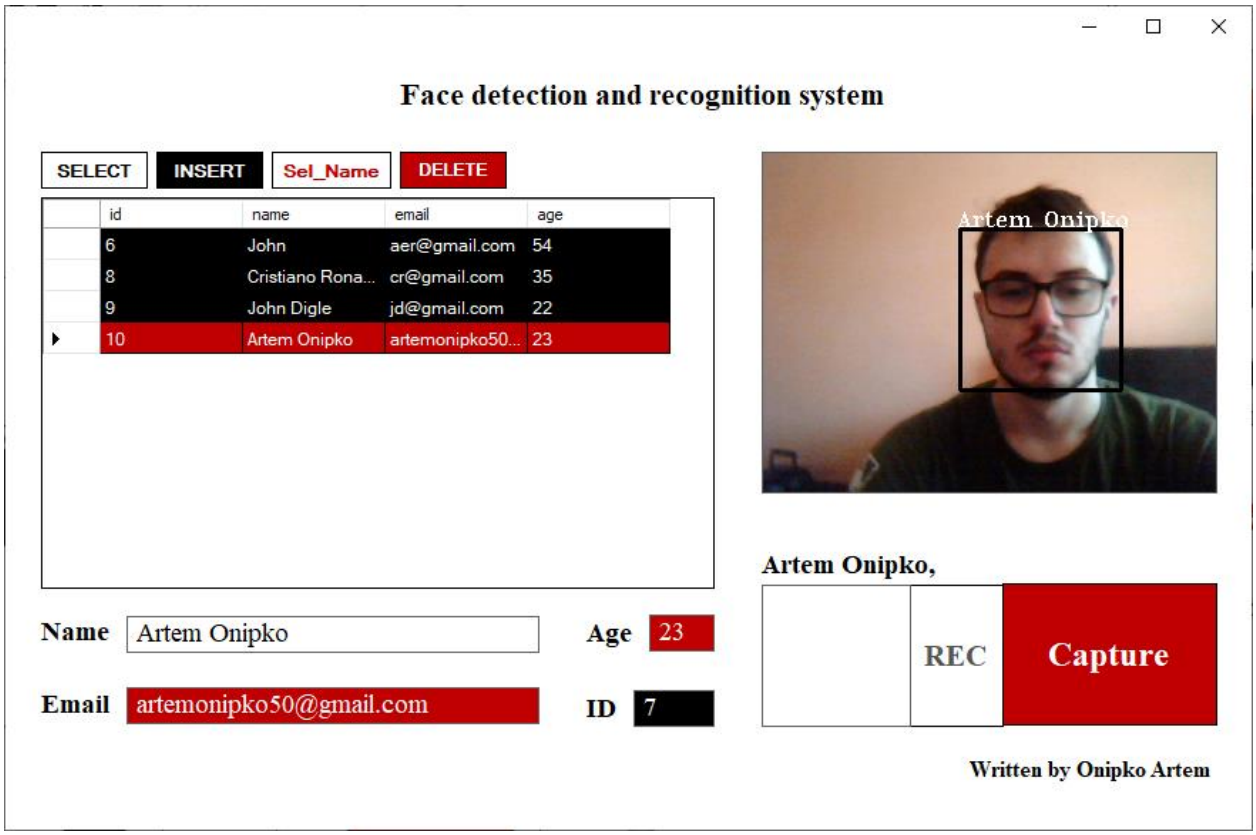
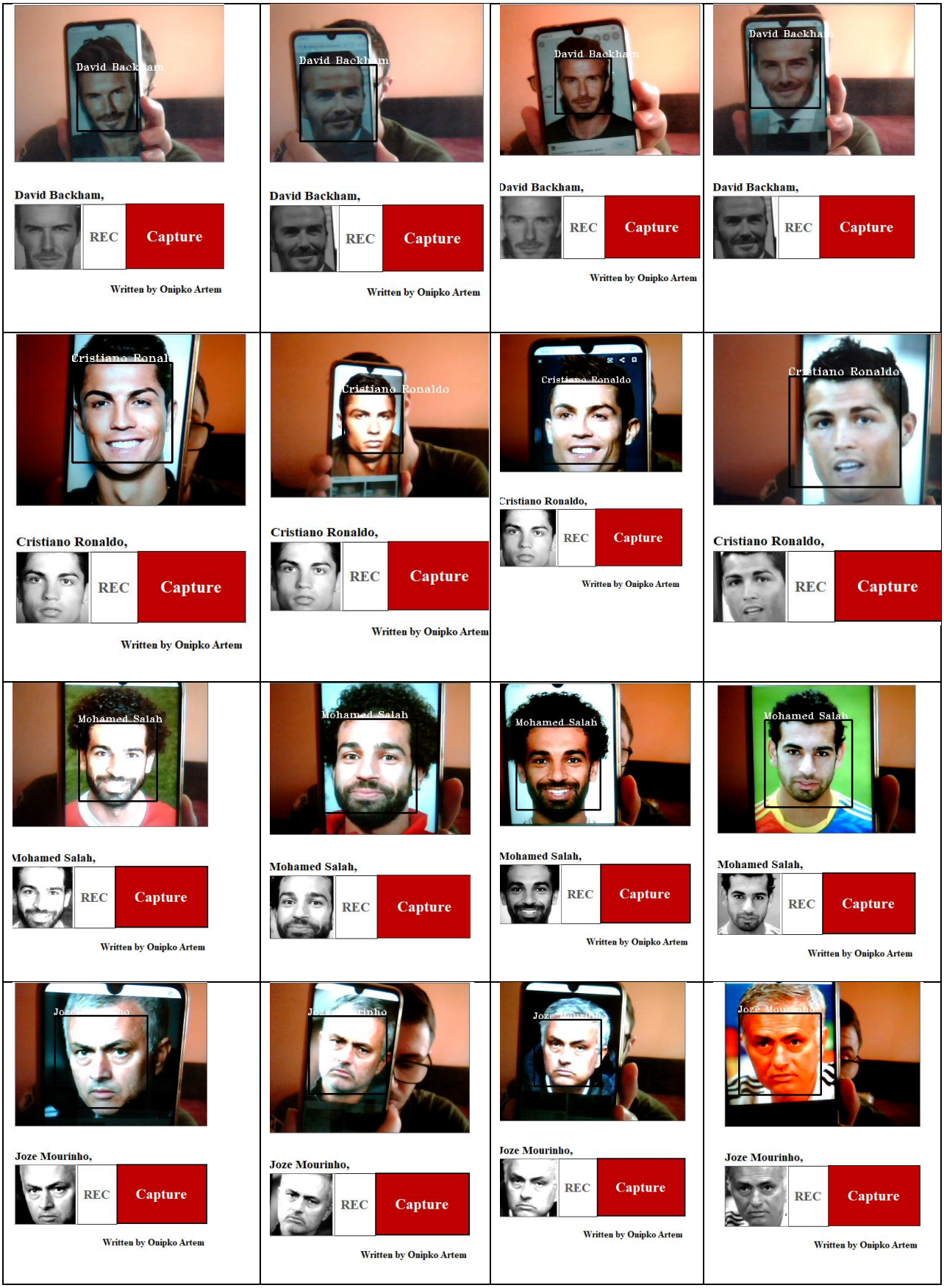


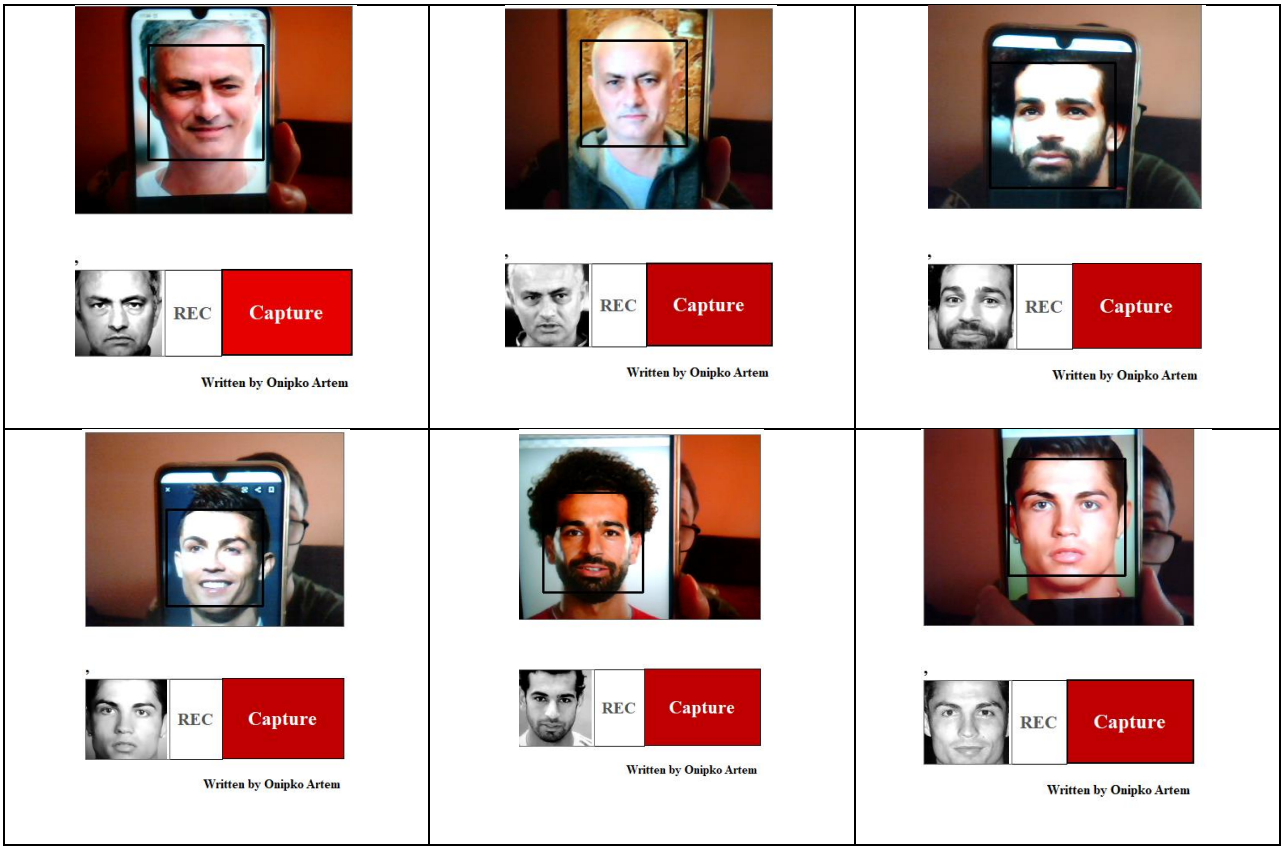
Рисунок 4.7. – детекція фото

Після натискання клавіши Capture система зберігає фото,таким чином відбувається навчання,тому чим більше даних щодо однією особи,тим вірогідність розпізнавання у наступному кроці підвищується. Аналіз розпізнавання різних людей за різними фото див. у таблиці 1.



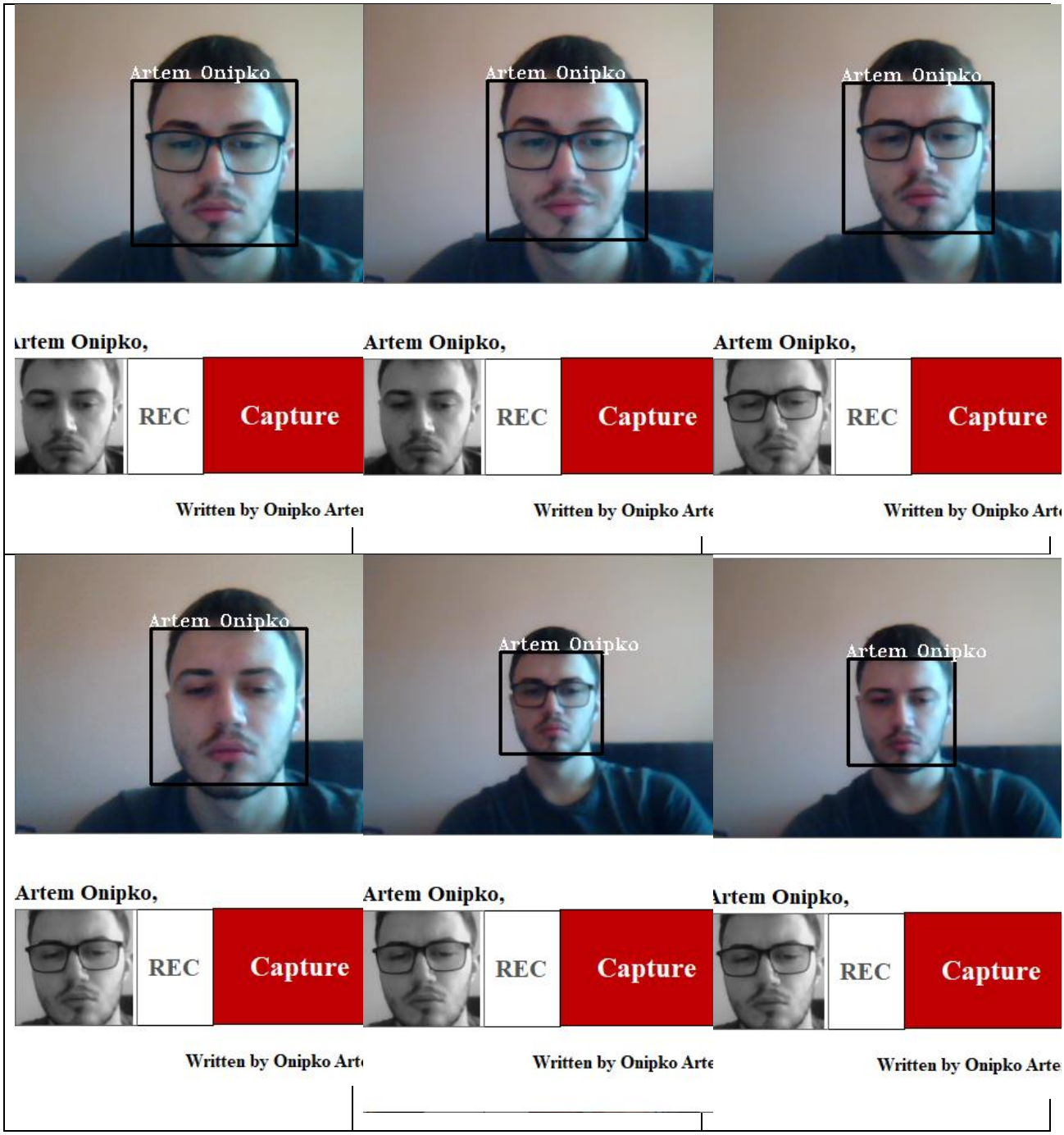
Таблиця 1 – приклади розпізнавання різних фото

Проблема даного застосування даної програми заключається у декілької проблемах,таких як навчання,яке потребує декількох фотографій одного типу під різними кутами, якість фотографії, умови в яких вона була зроблена – а це починаючи від гримаси, фону,шумів,кута нахилу і навіть різні фото однієї людини різного віку, Тому в наступній таблиці наведені помилки алгоритму.



Таблиця 2 – приклади помилок розпізнавання різних фото

Але даний алгоритм має переваги для розпізнавання людини у реальному часі,перешкодами для якого є мала кількість вхідних даних,оскільки вона може знадобитися для розпізнвання обличчя в тих самих умовах,але при різних кутах або гримасах особи.



Таблиця 3 – приклади розпізнавання особи в подібних умовах.

ВИСНОВОК

В даній роботі проведений аналіз детекції та розпізнавання обличчя, розглянуті сучасні приклади систем безпеки та приклади онлайн сервісів для виявлення осіб. Описаний стан речей у даній темі на сьогодення. Розглянуті методи які мали або мають застосування при детекції розрізнюванні облич,а також обрані конкретні саме для даного дипломного проекту. Після загального огляду алгоритмів які частково мають застосування у сучасних системах безпеки,була сформована постановка задачі, з якої було відомо подальші кроки написання документації а також допомога при формуванні розробки програмного забезпечення. Як метод детекції були обрані каскади за Хааром,як метод розпізнавання – це алгоритм знаходження обличчя за власною поверхнею. Для цих методів була сформована математична модель для подальшої алгоритмізації. В технічній документації був визначений опис даних для розробки програмного забезпечення. Насаперед це опис майбутньої бази даних, як додаток якої використовувався *MsSql Server, як середовище програмування і створення інтерфейсу було обране Microsoft Visual Studio 2019. Основою самого алгоритму стали бібліотеки OpenCV які значно полегшели роботи із побудовою програми, оскільки базові математичні алгоритми вже містяться у бібліотеках, подробиці можна знайти у розділі «Вибір програмної реалізації»*.Щодо самого алгоритму,то з налізу вибудовується висновок,що вірогідність розпізнавання іншої фотографії конкретного обличча при кількаразовій детекції однієї фотографії під різноманітними кутами складає <70%>, оскільки фотографії повинні бути зроблені в подібних умовах для того щоб алгоритм знайшов достантньо співпадінь. Умови стосуються отсвітлення,фону,мімики,наявності або відсутності окулярів тощо. Але при подібних умовах(а саме тих в яких зроблено фото,яке запам’ятав алгритм) алгоритм розпізнає особу онлайн з вірогідністю в 90%,похибка розпізнавання тієї ж людини залежать в основному від кута детекції,наявності/відсутності окулярів,освітлення тощо. В цілому розроблена система детеції та розпізнавання для виявлення осіб у кадрі, алгоритми працюють із взаємодією за базаю даних для перевірки коректності вводу користувача, який дає характеристики особі. В цілому інтерфейс,структура,даних,алгоритми та їх застосування у програмному середовищі були описані у розділі «Інформаційне та програмне забезпечення інтелектуальної інформаційної системи». Сама ж програма була протестована загальним чином, без поглиблення у теорію тестування,але дію програми та базове тестування можна побачити у Додатку Б,який також являє собою інструкцією користувача,для самої інсталяції був доданий Додаток Б. Отриманий досвід при написанні диплому дає розуміння у надійності і функціоналу деяких складових програмного забезпечення, а також важливість методологій розробки, для розпорядження часу та контролю ресурсами під час дипломного проекту.

СПИСОК ЛІТЕРАТУРИ

1. Habr.com - Анализ существующих подходов к распознаванию лиц. – https://habr.com/ru/company/synesis/blog/238129/
2. Studfile.net - Интегральное представление изображений. –https://studfile.net/preview /6234768/page:3/
3. Живрин Я.Э. Методы определения объектов на изображении / Я.Э. Живрин, Н.Б. Алкзир / Молодой ученый, 2018. – №7. – С. 8-19. – https://moluch.ru/archive/193/48447/
4. Oxozle - https://oxozle.com/2015/04/11/metod-raspoznavaniya-lic-violy-dzhonsa-viola-jones/ - Метод распознавания лиц Виолы-Джонса (Viola-Jones),Дмитрий Азаров.
5. Ushir A.U. MathematicalModeling for Face Recognition System / A.U. Ushir, A. A. Shete, S. N. Tiwar, V. R. Vishwakarma, M. Sanghavi // International Journal of Computer Application, 2013. – № 0975 (8887). – P. 13-16. – http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.735.9122&rep=rep1&type=pdf
6. Tanzeem Choudhury. Multimodal Person Recognition using Unconstrained Audio and Video / Brian Clarkson, Tony Jebara, Alex Pentland //Perceptual Computing Group MIT Media Laboratory Cambridge - http://www.cs.columbia.edu/~jebara/papers/TR-472.pdf
7. Dmitry Lagun. Detecting cognitive impairment by eye movement analysis using automatic classification algorithms //Cecelia Manzanares,Stuart M. Zola, Elizabeth A. Buffalo,Eugene Agichtein - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3403832/
8. Kevin Draper / Has Used Face-Scanning Technology on Customers - https://www.nytimes.com/2018/03/13/sports/facial-recognition-madison-square-garden.html
9. Василий Стефаник Прикарпатский / Методи Розпізнавання Облич: Короткий Огляд // кафедра інформаційних технологій ДВНЗ «Прикарпатський національний університет імені Василя Стефаника» - http://itcm.comp-sc.if.ua/2017/Holubiak.pdf.
10. Developer.com - https://developpaper.com/face-recognition-demo-parsing-c/ /Face recognition demo parsing c#.
11. Microsoft.com / Frank La Vigne / - Exploring Face Detection and Recognition / - https://docs.microsoft.com/en-us/archive/msdn-magazine/2019/november/artificially-intelligent-exploring-face-detection-and-recognition /

12. FoxLearn / Windows forms face detection and recognition using mgucv / - https://foxlearn.com/windows-forms/face-detection-for-net-using-emgu-cv-in-csharp-443.html
13. Medium.com / Find all faces that are looking directly at the camera with c# / Mark Farragher / - https://medium.com/machinelearningadvantage/find-all-faces-that-are-looking-directly-at-the-camera-with-c-59e8469696c1
14. Towardsdatascience / how to build a face detection and recognition system / Serhii Maksymenko / - https://towardsdatascience.com/how-to-build-a-face-detection-and-recognition-system-f5c2cdfbeb8c
15. Царьов Р.Ю. Біометричні технології: навч. посіб. [для вищих навчальних Ц18 закладів] / Р.Ю. Царьов, Т. М. Лемеха. – Одеса: ОНАЗ ім. О.С. Попова, 2016. – 140 с.: іл.

Додаток А

```
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Windows.Forms;
using Emgu.CV;
using Emgu.CV.Structure;
using Emgu.CV.CvEnum;
using System.IO;
using System.Diagnostics;
using System.Data.SqlClient;
using System.Data;
using System.Net.Mail;
using System.Net;
using System.Net.Mime;

namespace RecSystem
{
    public partial class FrmPrincipal : Form
    {
        //Об'явлення змінних, векторів і каскадів Хаара
        Image<Bgr, Byte> currentFrame;
        Capture grabber;  -  змінна для захвату
        HaarCascade face; - змінна для збереження каскадів обличчя за Хааром
        HaarCascade eye;  змінна для обчислення районів ока з допомогою каскадів Хаара
        MCvFont font = new MCvFont(FONT.CV_FONT_HERSHEY_TRIPLEX, 0.5d, 0.5d); - використання бібліотеки бля обчислення трикутників.
        Image<Gray, byte> result, TrainedFace = null; - змінна для виводу конвертування фото у байти
        Image<Gray, byte> gray = null;  - змінна для обчислення  конвертованого обличчя
        List<Image<Gray, byte>> trainingImages = new List<Image<Gray, byte>>();
        List<string> labels = new List<string>(); список для заповнення текстового документу імена обличь
        List<string> NamePersons = new List<string>(); список  для виводу імен під фото
        int ContTrain, NumLabels, t;
        string name, names = null;


        public FrmPrincipal()
        {
            InitializeComponent();
```

```
        //Завантаження каскадів для детекції та призначення до змінної face
        face = new HaarCascade("haarcascade_frontalface_default.xml");
        try
        {
            //Завантаження попереднього обличчя тренується поточне та мітки для кожного зображення
            string Labelsinfo = File.ReadAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt");
            string[] Labels = Labelsinfo.Split('/');
            NumLabels = Convert.ToInt16(Labels[0]);
            ContTrain = NumLabels;
            string LoadFaces;
            цикл для додавання ногово обличчя та збереження з конвертацією
            for (int tf = 1; trainedfaces < NumLabels + 1; trainedffaces++)
            {
                LoadFaces = "face" + tf + ".bmp";
                trainingImages.Add(new Image<Gray, byte>(Application.StartupPath + "/TrainedFaces/" + LoadFaces));
                labels.Add(Labels[tf]);
            }
        }
        catch (Exception e)
        {
            MessageBox.Show("Data base is not filled or changed,please add face to train.", "Trained faces is load",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }

    }



    //Конект до бази даних
    SqlConnection sqlconn = new SqlConnection("Data Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename="+
Application.StartupPath + "\\Database2.mdf;Integrated Security=True");


    private void button4_Click(object sender, EventArgs e)
    {
        if (!string.IsNullOrEmpty(textBox_age.Text) && !string.IsNullOrEmpty(textBox_name.Text) &&
!string.IsNullOrEmpty(textBox_email.Text))

        {
            sqlconn.Open();
            Запит щодо вибірки де вказана пошта,з таблиці
            SqlDataAdapter da = new SqlDataAdapter("select email from FACES where email ='" + textBox_email.Text + "'", sqlconn);
            DataTable dt = new DataTable();
```

```
    da.Fill(dt);
    sqlconn.Close();
    if (dt.Rows.Count >= 1)
    {
        MessageBox.Show("Duplicated email,email already exists", textBox_email.Text);
    }
    else
    {
        try
        {
            var addr = new System.Net.Mail.MailAddress(textBox_email.Text);

                sqlconn.Open();
                    запит щодо додавання  інформації до таблиці
                string sqlquery = "INSERT INTO [FACES] (name,age,email)VALUES(@name,@age,@email)";
                SqlCommand sqlcomm = new SqlCommand(sqlquery, sqlconn);
                int age = int.Parse(textBox_age.Text);
                sqlcomm.Parameters.AddWithValue("name", textBox_name.Text);
                if (age <= 100 && age > 18)
                {
                    sqlcomm.Parameters.AddWithValue("age", textBox_age.Text);
                    sqlcomm.Parameters.AddWithValue("email", textBox_email.Text);
                    sqlcomm.ExecuteNonQuery();
                    MessageBox.Show("Created Successfully ..");
                    sqlconn.Close();
                }
                else
                {

                    MessageBox.Show("Incorrect age input, value must be above 18 and below 100");
                    sqlconn.Close();
                }

            }
            catch (FormatException ex)
            {
                MessageBox.Show("Email type doesnt exist" + ex);
            }
    }
}
else
```

```
        {
            MessageBox.Show("Name,Age or Email are not filled!");
        }
    }


    private void button5_Click(object sender, EventArgs e)
    {
        string name = textBox_name.Text;
            вибірка з таблиці для подальшої обробки інформації
        string sqlquery = "SELECT * FROM FACES WHERE name LIKE '"+textBox_name.Text+"'";

        sqlconn.Open();
        SqlCommand sqlcomm = new SqlCommand(sqlquery, sqlconn);
            Заповнення таблиці
        SqlDataAdapter sdr = new SqlDataAdapter(sqlcomm);
        DataTable dt = new DataTable();
        sdr.Fill(dt);
        dataGridView1.DataSource = dt;
        sqlconn.Close();
    }


            Обробка для виділення діапазону правильних даних при вводі віку
    private void textBox_age_TextChanged(object sender, EventArgs e)
    {
        int result;
        if (textBox_age.Text != "")
        {
            if (!int.TryParse(textBox_age.Text, out result))
            {
                textBox_age.Text = "";
                MessageBox.Show("Invalid Integer");
            }
        }
    }


Обробка для виділення діапазону правильних даних при вводі айді
    private void textBox_id_TextChanged(object sender, EventArgs e)
    {
```

```
            int result;
            if (textBox_id.Text != "")
            {
                if (!int.TryParse(textBox_id.Text, out result))
                {
                    textBox_id.Text = "";
                    MessageBox.Show("Invalid Integer");
                }
            }
        }



        //видалення інформації з таблиці
        private void button6_Click(object sender, EventArgs e)
        {
            if (!string.IsNullOrEmpty(textBox_id.Text))
            {
                sqlconn.Open();
вибірка з таблиці за айді для подальшого виділення
                SqlDataAdapter da = new SqlDataAdapter("select * from FACES where id ='" + textBox_id.Text + "'", sqlconn);
                DataTable dt = new DataTable();
                da.Fill(dt);
                sqlconn.Close();
                if (dt.Rows.Count == 0)
                {
                    MessageBox.Show("Person is not found,because id does not exist", textBox_id.Text);
                }
                else{
                        sqlconn.Open();
                        string sqlquery = "delete from FACES where id = '"+textBox_id.Text+"'";
                        SqlCommand sqlcomm = new SqlCommand(sqlquery, sqlconn);



                        sqlcomm.ExecuteNonQuery();
                        MessageBox.Show("Person deleted Successfully ..");
                        sqlconn.Close();
                    }
            }
            else
            {
                MessageBox.Show("Id is not filled!");
```

```
        }
    }



    //пошук Ім'я в таблиці
    private void button3_Click_1(object sender, EventArgs e)
    {
        string sqlquery = "select *from [FACES]";

        sqlconn.Open();
        SqlCommand sqlcomm = new SqlCommand(sqlquery, sqlconn);

        SqlDataAdapter sdr = new SqlDataAdapter(sqlcomm);
        DataTable dt = new DataTable();
        sdr.Fill(dt);
        dataGridView1.DataSource = dt;
        sqlconn.Close();
    }
            Кнопка для захоплення картинки
    private void button1_Click(object sender, EventArgs e)
    {
        //Ініціалізація девайсу
        grabber = new Capture();
        grabber.QueryFrame();
        //Ініціалізація захвату
        Application.Idle += new EventHandler(FrameGrabber);
        button1.Enabled = false;
    }



Кнопка для додавання
    private void button2_Click(object sender, System.EventArgs e)
    {
        if (!string.IsNullOrEmpty(textBox_name.Text))
        {
            sqlconn.Open();
            SqlDataAdapter da = new SqlDataAdapter("select email from FACES where name ='" + textBox_name.Text + "'",
sqlconn);
            DataTable dt = new DataTable();
            da.Fill(dt);
```

```
            sqlconn.Close();
            if (dt.Rows.Count == 0)
            {
                MessageBox.Show("To add new face, make sure that information about this person's name(is in textfield name) already exists in table, if it does not, you can create another one using button INSERT above table", textBox_email.Text);
            }
            else
            {

                try
                {
                    //Навчений лічильник
                    ContTrain = ContTrain + 1;

                    //Сірий кадр від пристрою захоплення
                    gray = grabber.QueryGrayFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);

                    //Детектування за допомогою каскадів Хаара
                    MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(face,1.2,10,Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
                    new Size(20, 20));

                    // Дія для кожного виявленого елемента
                    foreach (MCvAvgComp f in facesDetected[0])
                    {
                        TrainedFace = currentFrame.Copy(f.rect).Convert<Gray, byte>();
                        break;
                    }

                    //змінити розмір виявленого зображення на обличчі того, щоб порівняти той самий розмір з попереднім
                    //тестове зображення методом кубічної інтерполяції
                    TrainedFace = result.Resize(100, 100, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
                    trainingImages.Add(TrainedFace);
                    labels.Add(textBox_name.Text);

                    //Показати обличчя в області
                    imageBox1.Image = TrainedFace;

                    //Кількість облич показана в області,котрі збережені у текстовому файлі
                    File.WriteAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt", trainingImages.ToArray().Length.ToString() + "/");
```

```
                for (int i = 1; i < trainingImages.ToArray().Length + 1; i++)
                {
                    trainingImages.ToArray()[i - 1].Save(Application.StartupPath + "/TrainedFaces/face" + i + ".bmp");
                    File.AppendAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt", labels.ToArray()[i - 1] + "/");
                }

                MessageBox.Show(textBox_name.Text + "´s face detected and added", "Training OK", MessageBoxButtons.OK,
MessageBoxIcon.Information);
            }
            catch
            {
                MessageBox.Show("Enable the REC first", "Training Fail", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
        }
    }
    else
        MessageBox.Show("Fill textfield name");
}



void FrameGrabber(object sender, EventArgs e)
{
    label3.Text = "0";
    //label4.Text = "";
    NamePersons.Add("");


    // Інформація збору форми поточного кадру
    currentFrame = grabber.QueryFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);

        //Конвертування до сірогї відтінків Grayscale
        gray = currentFrame.Convert<Gray, Byte>();

        //Детектування обличчя
        MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(
        face,
        1.2,
        10,
        Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
        new Size(20, 20));
```

```
            //Дія для кожного елементу
            foreach (MCvAvgComp f in facesDetected[0])
            {
               t = t + 1;
               result = currentFrame.Copy(f.rect).Convert<Gray, byte>().Resize(100, 100,
Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
               //змалювання обличчя, виявлене в 0-му (сірому) каналі, синім кольором
               currentFrame.Draw(f.rect, new Bgr(Color.Black), 2);



               if (trainingImages.ToArray().Length != 0)
               {
               //Терміни Критерії розпізнавання обличчя з кількістю тренованих зображень, таких як maxIteration
               MCvTermCriteria termCrit = new MCvTermCriteria(ContTrain, 0.001);

                 //Розпізнавальник особи власного обличчя
                 EigenObjectRecognizer recognizer = new EigenObjectRecognizer(
                 trainingImages.ToArray(),
                 labels.ToArray(),
                 3000,
                 ref termCrit);

               name = recognizer.Recognize(result);

               //змалювання мітки для кожного виявленого та розпізнаного обличчя
               currentFrame.Draw(name, ref font, new Point(f.rect.X - 2, f.rect.Y - 2), new Bgr(Color.White));
               }

                  NamePersons[t-1] = name;
                  NamePersons.Add("");



               //кількість облич, виявлених на сцені
               //label3.Text = facesDetected[0].Length.ToString();



            }
               t = 0;

               //Ім'я розпізнанних облич
```

```
                for (int nnn = 0; nnn < facesDetected[0].Length; nnn++)
                {
                    names = names + NamePersons[nnn] + ", ";
                }
                //Вивід розпізнаних облич
                imageBoxFrameGrabber.Image = currentFrame;
            // лейбл у який виводиться ім'я
                label4.Text = names;i
                names = "";
                //Очистити лейбл
                NamePersons.Clear();

        }
    }
}
```

Додаток Б

У цьому додатку проведене загальне тестування програми та містить інструкції користувача. Для початку роботи роботи алгоритмів потрібно натиснути кнопку **REC**. Після цього ,якщо виконані умови для розпізнавання та/або детекції(а саме світло,кут нахилу,наявність або відсутність даних), у вікні справа рис Б.1 обличчя буде або ідентифікована як обличчя,або як обличчя особи.

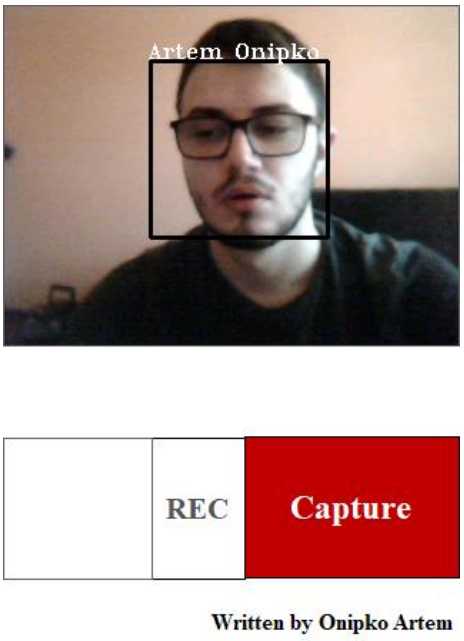


Рисунок Б.1. – приклад детекції та ідентифікації особи після натиснутої конопки **REC**

Користуванч може зберегти обличчя для подального його ідентифікування. Ям метод збереігання обличчя є кнопка Capture,але для збереження фото інформація про особу має міститися у базі даних справа за здалегідь,а 57м.'я у полі **Name** відповідати імені у таблиці,вибірку якої можна виконати за допомогою клавішу **SELECT** рис. Б.2.

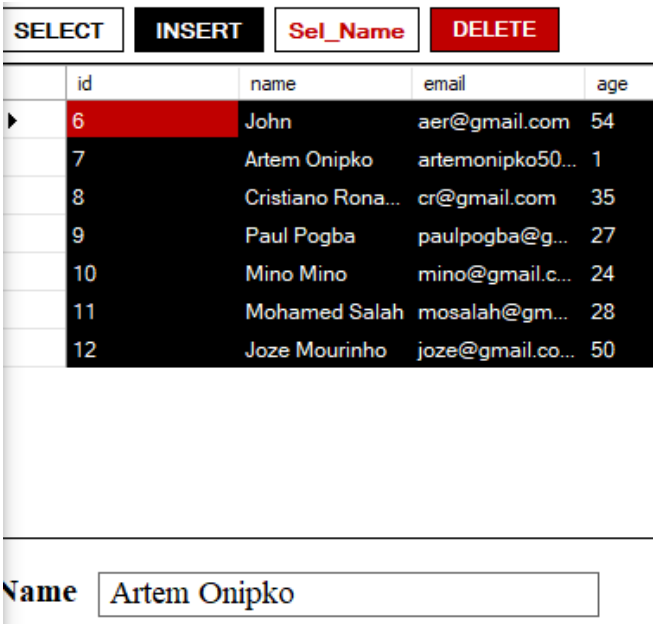


Рисунок Б.2. – таблиця з інформацією осіб.

При умові якщо користувач лишив поле пустим і натиснув кнопку **Capture**,виникне помилка рис.Б.3.

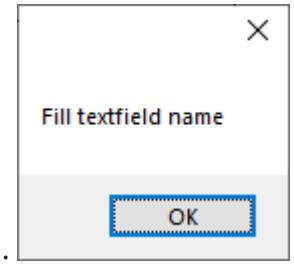


Рисунок Б.3 – помилка пустої строки.

Відповідно якщо користувач заповнив поле **Name,**але при цьому 58м.'я не відповідає імені з таблиці,тоді виникатиме помилка про відсутність інформації рис Б.4.

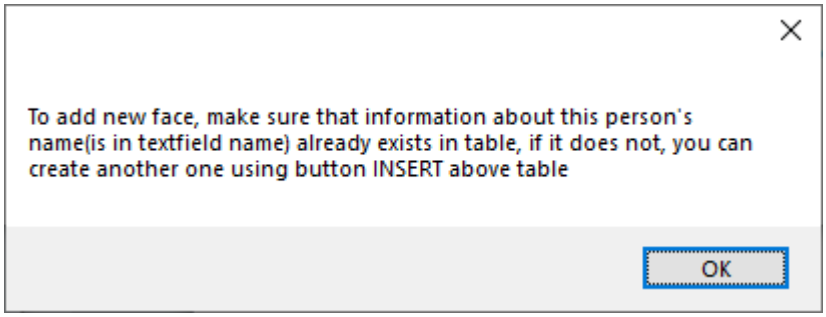


Рисунок Б.4. – помилка про відсутність інформації про особу

Для того щоб переконатися у відповідності інформації, зверху таблиці є клавіша **Sel_Name,** натиснувши яку,користувач може переконатися у наявності або відсутності інформації про особу,фото якої він хоче додати,для подальшого розпізнавання(рис.Б.5 – Б.6), а також клавіша **Select** яка вибирає здійснює вибір всіх даних.

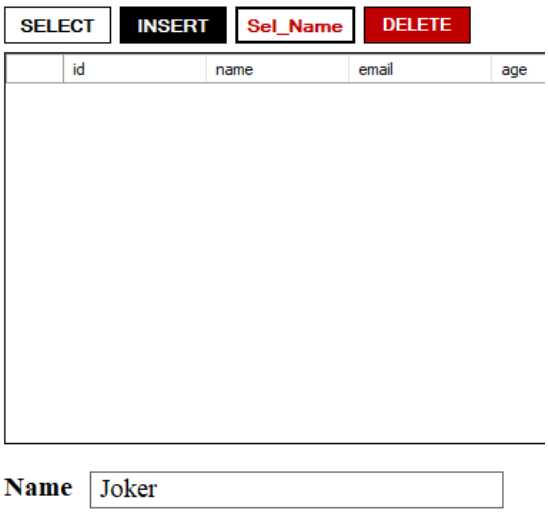


Рисунок Б.5.- пошук при відсутності даних	Рисунок Б.6.- пошук при наявності даних

Якщо користувач потребує новис запис у таблицю,він має змогу заповнити поля **Name,Age,Email** і натиснути після цього клавішу **Insert**, якщо користувач ввів значення вірним чином,то інформація буде додана до бази із подібним сповіщенням(рис. Б7), і фото людини з таблиці можна зберегти за допомогою клавіши **Capture**(рис. Б.8).

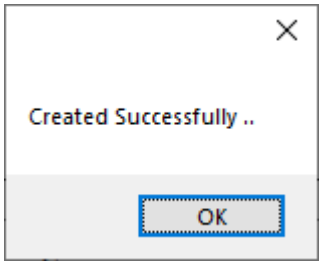


Рисунок Б.7. – вікно сповіщення про створення нової інформації .

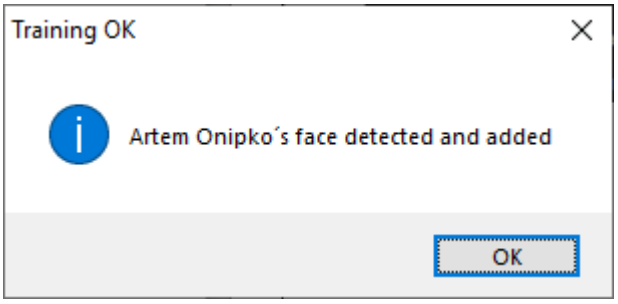
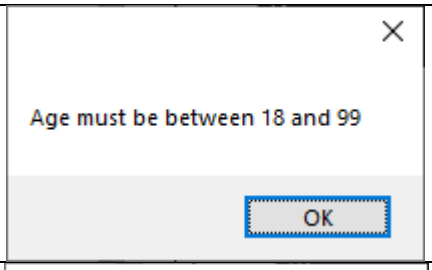
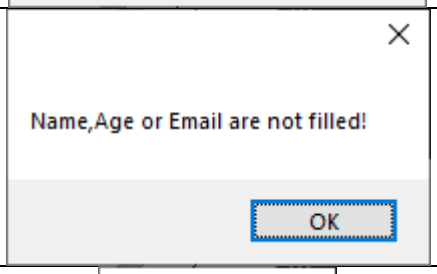
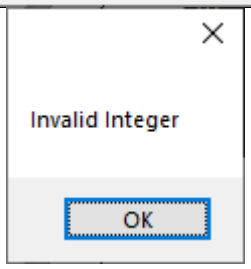
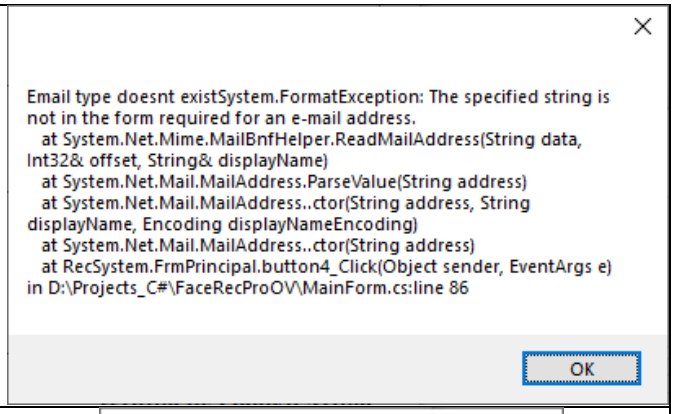
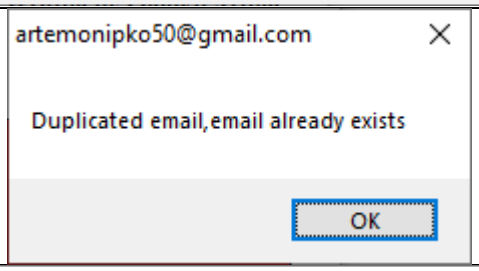
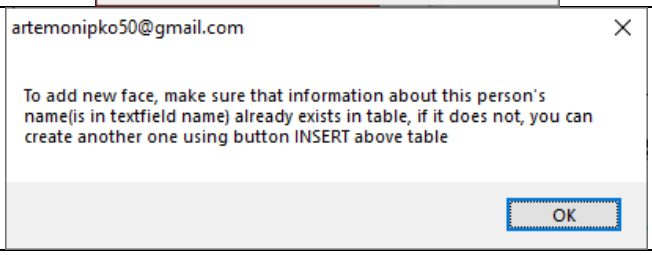


Рисунок Б.8. – вікно сповіщення про додавання обличчя до бази.

Але якщо користувач вводить у поля недопустиму інформацію як(Дубльована пошта або не відповідний тип у полі Email,неправильне значення у полі Age або введення невірного типу даних,як приклад String або Double,або відсутність заповнення одного або декількох полів),тоді користувач може отримувати наступні помилки (таблиця Б.1.)

З'вляється при умові якщо користувач,вводить вікове значення за діапазоном від 18 до 99 у поле **Age**.	
Виникає при умові відсутності заповнення одного або декілька полів.	
Дана помилка спричинена не вірним вводом у поле **Age**. А саме ввод алфавітних букв або знаків з клавіатури.	

Дане вікно повідомляє про те що користувач ввів тип пошти який не існує у поле **Email**.	
Повідомлення про введення користувачем пошти у поле **Email** яка вже існує у таблиці.	
Виникає при спробі користувача додати нове обличчя,після того як було введене 60м.'я у поле **Name**, яке не зберігається у таблиці і натиснута клавіша **Capture.**	

Таблиця Б.1 можливі помилки через невірний ввід користувачем.

Якщо ж додав інформацію вірно у таблицю,і хоче додати фото до бази,користувач має змогу натиснути **Capture**,при умові якщо поле Name не пусте.Програма сповістить про успішне додавання. (Рисунок – Б.9.)

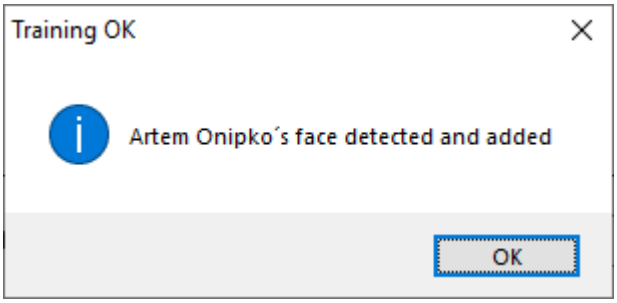


Рисунок – Б.9.

Додаток В

Інструкція інсталяції

Для інсталювання програмного продукту требя відкрити файл «Setup» і натиснути кнопку «Далі» (Рис. В.1).

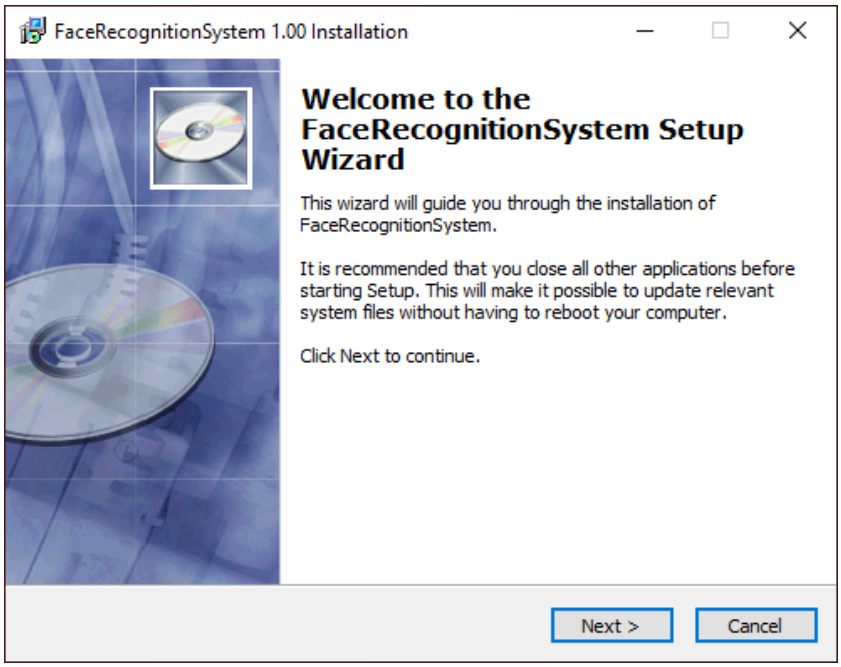


Рисунок В.1 – Перша сторінка ісінталяції

Для зміни шляху встановлення є можливість змінити адресу директорії вручну або натиснути кнопку огляд та скористатись провідником.(Рис. В.2) Після цього необхідно натиснути копку «Next»

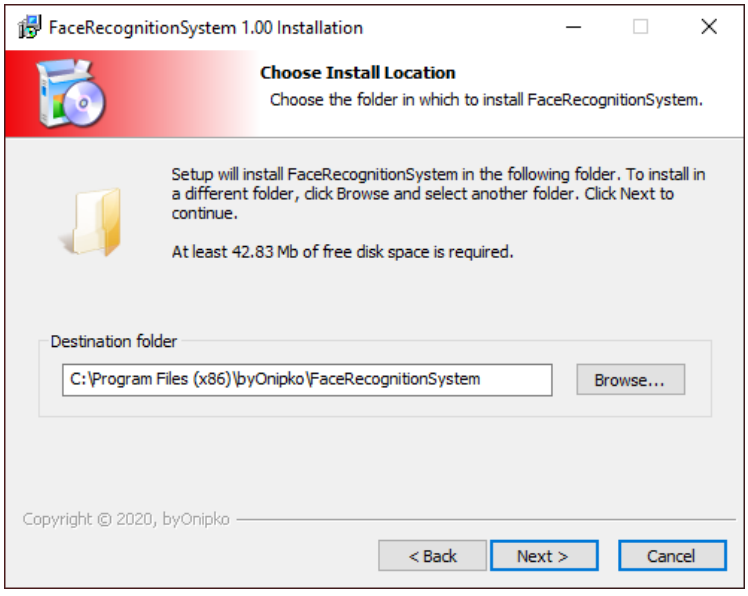


Рисунок В.2 – Вибір шляху встановлення ПЗ

В передостанньому вікні є можливість переглянути всі обрані параметри при виконанні інсталяції. Для початку ісінталяції необхідно натиснути кнопку «Install»(Рис. В.4)

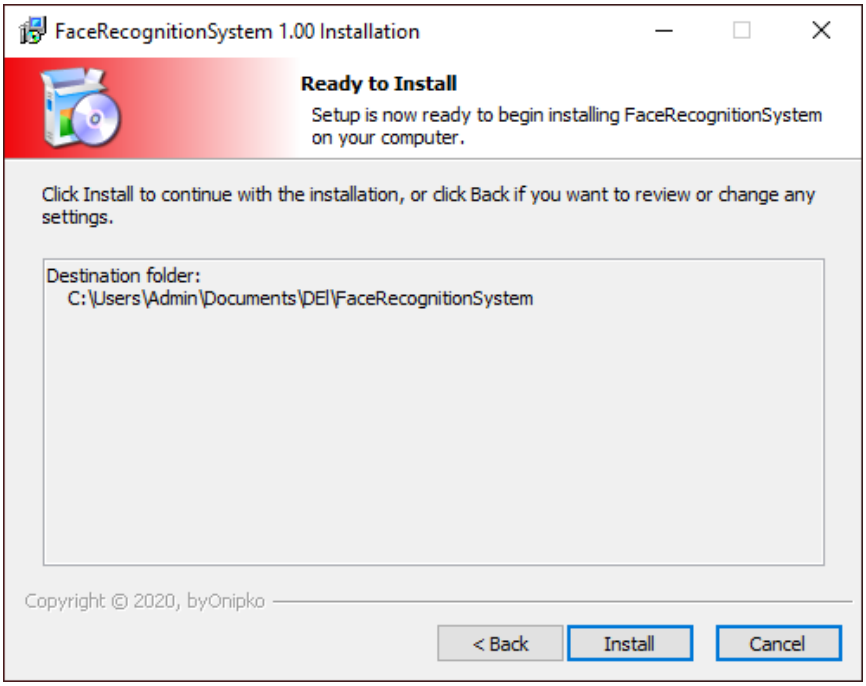


Рисунок В.3 – Вікно перед початком інсталяції

В останньому вікні є можливість переконатись у корректності виконанної інсталяції. Для завершення інсталяції необхідно натиснути кнопку «finish».(Рис.В.5)

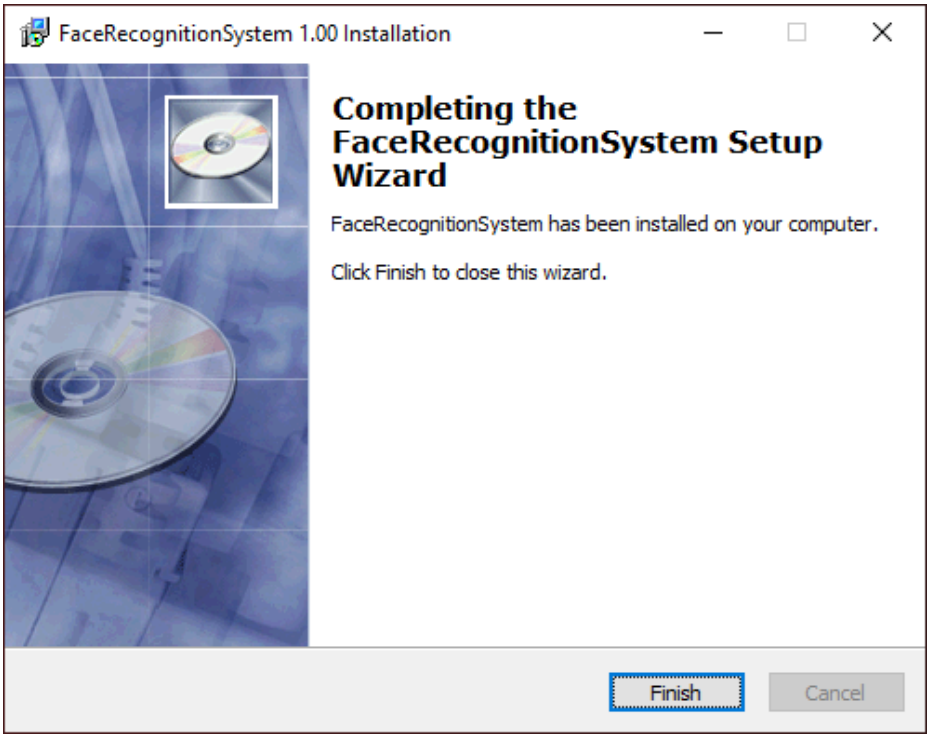


Рисунок В.4 – Вікно завершення інсталяції