НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Факультет прикладної математики

Кафедра прикладної математики

До захисту допущено

Завідувач кафедри ПМА

_____О. Р. Чертов

«___»_____2015р.

Дипломна робота

на здобуття ступеня бакалавра

зі спеціальності 6.040301 «Прикладна математика»

на тему: Програмний засіб для перемикання активного вікна за фокусуванням погляду для ОС Windows

Виконав: студент 4 курсу, групи КМ-11

Хижняк Владислав Юрійович _____

Керівник: к.т.н, доцент Сирота С.В. _____

Консультант з нормоконтролю:

старший викладач Мальчиков В.В. _____

Рецензент: доцент кафедри системного проектування ННК ІПСА НТУУ «КПІ» доцент Цурін О.Ф. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ – 2015 року

Національний технічний університет України

«Київський політехнічний інститут»

Факультет прикладної математики

Кафедра прикладної математики

Рівень вищої освіти – перший (бакалаврський)

Спецільність 6.040301 «Прикладна математика»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____О. Р. Чертов

«___»_____2015р..

ЗАВДАННЯ

на дипломну роботу студенту

Хижняку Владиславу Юрійовичу

1. Тема роботи «Програмний засіб для перемикання активного вікна за фокусуванням погляду для ОС Windows»

керівник роботи – к.т.н Сирота С.В.

затверджені наказом по університету від ”19”травня 2015 р. № 1039-С.

2. Строк подання студентом роботи “12” червня 2015р.

3. Вихідні дані до роботи:

- Зображення з повним обличчям користувача в сірих тонах;
- Необхідні шаблони з якірними точками;

4. Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити):

- Вивчити літературні джерела за тематикою комп’ютерного зору;
- Провести аналіз існуючих програмних рішень у даній галузі;
- Проаналізувати математичне забезпечення, що використовується в комп’ютерному зорі;

- Розробити математичну модель, яка визначає вектор напряму погляду за даними, що були отримані з вебкамери;
- Програмно реалізувати та протестувати дану модель;
- Розробити резидентну програму для ОС Windows, що переимкає активне вікно в залежності від напрямку погляду;
- Оформити документацію до дипломної роботи;

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень):

- Архітектура програмного забезпечення;
- Структурна схема програми;
- Взаємодія роботи підсистем;
- Екранні форми програми.

6. Консультанти розділів проекту (роботи):

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	ст.викладач Мальчиков В.В.		

7. Дата видачі завдання «28» жовтня 2013

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Строк виконання етапів роботи	Примітка
1.	Вивчення літератури за тематикою роботи	13.11.14	
2.	Проведення порівняльного аналізу математичних методів	2.12.14	
3.	Вибір, обґрунтування та опановування методів розв‘язку задачі	8.01.15	
4.	Проектування архітектури розроблюваних програмних засобів	15.01.15	
5.	Визначення складу та форматів вихідних даних та результатів для кожної програми	4.02.15	
6.	Розробка алгоритмів	18.02.15	
7.	Програмна реалізація	9.04.15	
8.	Розробка алгоритмів та підготовка контрольних задач для їх перевірки	23.04.15	
9.	Розв‘язування контрольних задач на ПЕОМ	13.05.15	
10.	Оформлення документації дипломної роботи		

Студент _____ Хижняк В.Ю.

Керівник роботи _____ Сирота С.В.

АНОТАЦІЯ

Дана дипломна робота виконана на здобуття освітньо-кваліфікаційного рівня “Бакалавр” та присвячена розробці програмного засобу для перемикання активного вікна за фокусуванням погляду на ОС Windows.

Метою роботи є створення програмного забезпечення, яке буде відстежувати фокусування погляду людини та перемикати активне вікно в залежності від монітору, на який користувач фокусує свій погляд.

В рамках дипломної роботи проведено аналіз існуючих математичних методів, що виконують схожі задачі. Були виділені критерії, на основі яких обирався основний метод роботи програми.

В роботі реалізовано математичну модель відстеження фокусування погляду людини, яка базується на основі модифікованого методу збігу за шаблонами та модифікованої інтегральної матриці. Для визначення активного монітору використовується пошук точки перетину вектору фокусування погляду людини та моделі монітору.

Отримані результати можуть бути використані при створенні інтелектуальних систем керування у різних предметних областях.

Робота виконана на 100 аркушах, має посилання на список використаних літературних джерел з 23 найменування, а також наведено 18 рисунків, 5 таблиць та 2 додатки.

Ключові слова: управління очами, метод пошуку за шаблоном, переваги, перспективна інверсія за трьома точками, розпізнавання об’єктів, комп’ютерний зір.

ABSTRACT

This diploma is carried out to obtain educational qualification of "Bachelor" and devoted to developing software tool to switch an active window in Windows OS .

The target is to create eye gaze tracking and analysing software. Obtained data can be used for switching active windows depending on a monitor that is gaze focused by user.

As a part of the thesis, an analysis of existing math methods that resolve similar tasks was done. Criteria which were elected the general method of the program were identified.

In this paper, a mathematical model of eye gaze tracking that is based on modified template matching and integrated matrix was implemented. To determine the active screen there is used the focus point of intersection of the gaze vector and a monitor model.

These results can be used to create intelligent control systems in various subject areas.

Work carried out on 100 pages, contains references to the list of used literature of 23 names, and provides 18 figures, 5 tables and 2 appendices.

Keywords: eye control, template matching, benefits, perspective inversion for three points, object recognition, computer vision.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

ВФП (Відстеження фокусування погляду) — процес визначення вектору напряму погляду чи руху ока відносно голови.

Комп'ютерний зір або Комп'ютерне бачення (Computer Vision) — теорія та технологія створення машин, які можуть проводити виявлення, стеження та класифікацію об'єктів.

МП (від англ. Megapixel) — один мільйон (1 000 000) пікселей, що формують зображення.

AAM (скорочено від англ. Active Appearance Models) — це статистичні моделі форми та зовнішнього вигляду об'єктів, які можуть багаторазово деформуватися, для підгонки до об’єкта, беручи до уваги явні обмеження форми.

ABM (скорочено від англ. Appearance Based Methods) — методи пошуку об’єктів, що базуються на “зовнішньому” вигляді об’єктів зображення.

ASM (скорочено від англ. Active Shape Models) — це статистичні моделі форми об'єктів, які можуть багаторазово деформуватися, для підгонки до об'єкта, присутньому на новому зображенні.

FBM (скорочено від англ. Feature Based Methods) — методи пошуку об’єктів, що базуються на особливих точках зображення.

OpenCV (скорочено від англ. Open Source Computer Vision Library, бібліотека комп'ютерного зору з відкритим вихідним кодом) — бібліотека функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом.

RGB (скорочено від англ. Red, Green, Blue — червоний, зелений, синій) — адитивна колірна модель, що описує спосіб синтезу кольору, за якою червоне, зелене та синє світло накладаються разом, змішуючись у різноманітні кольори.

TM (скорочено від англ. Template Matching) — техніка цифрової обробки зображень для знаходження дрібних деталей зображення, які відповідають шаблон-зображенню.

UML (скорочено від англ. Unified Modeling Language) — уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення.

VJA (скорочено від англ. Viola-Jones Algorithm) — алгоритм, що дозволяє виявляти об'єкти на зображеннях в реальному часі.

ВСТУП

В нашому житті все більше різноманіття електронних пристроїв. Вони становляться частиною всього, що оточує нас і інтегрується в повсякденне життя. В зв'язку з цим приходить необхідність працювати з ними для поліпшення буденних справ, як то навчання, робота, спілкування, наукова діяльність, тощо.

Вже придумано багато механічних засобів взаємодії з пристроями, як то клавіатура, миша, навігаційні клавіші, пульти керування, сенсорний екран та інше. Однак, в останній час набирає популярність безконтактна взаємодія: за допомогою голосу чи жестів, як це стверждують та розділяють думку автори таких творів [1] та [2].

Менш популярним але більш багатообіцяючим є безконтактна взаємодія за допомогою очей. Це пов'язано з тим, що людині, що працює на комп'ютері, треба менше часу для того щоб зробити дію на комп'ютері за допомогою просто зору ніж додати до цього ж механічну дію – перетаскування курсору за допомогою допоміжних пристроїв. Також такий тип взаємодії піде на добру службу людям з обмеженими можливостями, як зазначив Бласко у своїй роботі[3].

Група авторів з журналу з нейроінжинірингу описують у своїй праці [4], що ВФП використовується в багатьох галузях, таких як психологія, когнітивна лінгвістика та інші. Особливе місце ВФП завоювала в маркетингу для тестування реклами.

Одні з перших моделей ВФП вимагали фіксації голови респондента або ж модель ВФП закріплювалася на голові. Подалі було розроблено моделі ВФП на основі декількох камер разом з пристроями, що випромінюють інфрачервоними променями, як це робили І. Сігут та П. Сітха у своїй роботі [5], або LED ліхтариками, що утворюють якусь просту геометричну фігуру (коло, квадрат, зірка, тощо), як це робили П. Роланд та Х. Хуа [6]. Згідно звітам за цими

роботами, все це дозволяло з високою точністю прорахувати фокусування погляду респондента але в той час, як зазначив С. Вестон у своїй праці на конференції, присвяченій впливу людини на комп'ютер та способи їх взаємодій [7], не надавало можливості використовувати ВФП на звичайних робочих системах, так як вимагало додаткового обладнання.

У даній роботі розглядається можливість ВФП за допомогою камери з невеликою роздільною здатністю на системі з декількома моніторами. Такі камери встановлені на більшості робочих систем, тобто не потребує додаткових пристроїв для налаштування роботи програми на основі ВФП.

Задача зафіксувати та відстежити напрям погляду людини на робочій станції з вбудованою камерою з роздільною здатністю від 1 МП поставлена і успішно розв'язана.

Ця задача не є елементарною для комп'ютера, тому саме потребується від користувача зазначеної вище системи виконання певних передумов для її правильної роботи.

Вирішення задачі складається з декількох підзадач:

1. Знаходження лиця користувача за допомогою камери.
2. Приведення голови до 3D моделі.
3. Позиціонування голови відносно камери.
4. Знаходження очей на обличчі користувача.
5. Знаходження положення зіниці на оці.
6. Трасування фокусування погляду.

Ця робота є початковим етапом до подальшого розвитку програми у направлені комп'ютерного зору та інтерактивних систем. Подалі планується побудувати систему, що дозволить управляти комп'ютером за допомогою очей та жестів.

МЕТА ДОСЛІДЖЕННЯ

Провести аналіз сучасних методів обробки зображень, відеопотоку, тощо.

Створити представлення роботи програми, що розпізнає об'єкти та складові голови людини, а саме око людини, лице людини, тощо.

Ознайомитись з існуючими програмними засобами комп'ютерного зору, що виконують подібну функцію.

Ознайомитися з програмними бібліотеками, що використовуються цим програмним забезпеченням.

Розробити математичну модель, яка визначає вектор напряму погляду за даними, що отримані з веб-камери.

Програмно реалізувати протестувати дану модель.

Розробити прототип програми, який може бути використаний для систем, що використовують ВФП, та для подальших досліджень.

ПОСТАНОВКА ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

Основною задачею дипломної роботи є створення програмного засобу для перемикання активного вікна за фокусуванням погляду на ОС Windows.

Допоміжним приладом для перевірки є вбудована камера з роздільною здатністю 1 МП.

При виконанні зазначених нижче умов система повинна знаходити особливі точки області очей людини та оперувати цими даними для моніторингу фокусування погляду людини:

- Камера повинна фіксувати обидва ока, тобто користувач має знаходитися перед камерою таким чином, що коли він фокусує погляд, камер має можливість зафіксувати обидва ока.
- Камера має фіксувати лице користувача повністю, має бути достатнє освітлення для виділення границь та форми лиця та його складових (ніс, очі, рот, тощо).
- Відстань від камери до користувача має бути не більше 1.0 м.
- Лице не має бути деформовано.
- Ніякий інших об’єктів, що не належать до частин обличчя людини, не повинно бути на фоні обличчя.
- Обличчя має знаходитися приблизно в вертикальному положенні відносно камери.
- Монітори повинні знаходитися в фіксованому положенні відносно камери під час роботи програми.
- Користувач задає положення моніторів з допомогою процедури калібровки.

Система моніторингу повинна видати користувачу проміжну інформацію про положення очей та вектор напряму погляду з кожного ока.

Система моніторингу в подальшому планується використовуватися в програмних засобах, що взаємодіють з користувачем напряму. Тому при виборі методів рішення задачі використовуються наступні критерії:

1. Мають бути використані найоптимальніші за швидкістю та розміром споживної пам'яті методи з існуючих на сьогоднішній день.
2. Обрані методи мають бути прості для сприйняття та підготовки до реалізації.
3. Має бути налагоджена обробка не менше десяти фреймів на секунду.
4. Усі знайдені особливі точки мають бути відображені на фреймі.
5. Для тестування програми необхідно передбачити можливість виводити усі здобуті дані про особливі точки на обличчі людини.

Програмна модель має бути реалізована на операційній системі Windows8.1.

1 ОГЛЯД І АНАЛІЗ МЕТОДІВ РІШЕННЯ

Для вирішення поставленої задачі вже існують багато методів та алгоритмів. Усі вони були створені для вирішення проблем і задач комп'ютерного зору.

В частині 1.1 комп'ютерний зір розглядається як наукова дисципліна, її історія та сфера використання.

Так як поставлена задача є комплексною, то для її спрощення розглядаються такі підзадачі:

- Пошук частин обличчя людини, такі як: ніс, очі, рот. Пошук вектору напряму голови людини.
- Пошук особливих точок ока, такі як: зіниця, кути очей.
- Пошук вектору напряму фокусування погляду ока та знаходження точки перетину з областю монітора.

Для того щоб знайти вектор напряму голови людини, а в подальшому і її вектор фокусування погляду необхідно знати положення голови в просторі.

Шість позиційних параметрів зберігають усю необхідну інформацію про положення та напрям голови в просторі. Для положення голови використовуються значення по трьом вимірам (абсциса, ордината та апліката).

Інші три ступені свободи позначають обертальний момент навколо кожної координати (кут процесії, кут власного обертання та кут нутації).

Головна ціль на цьому етапі – це знаходження шість ступенів свободи з одного зображення, отриманого з однієї камери та яка фіксує положення користувача, як показано на рисунку 1.1.

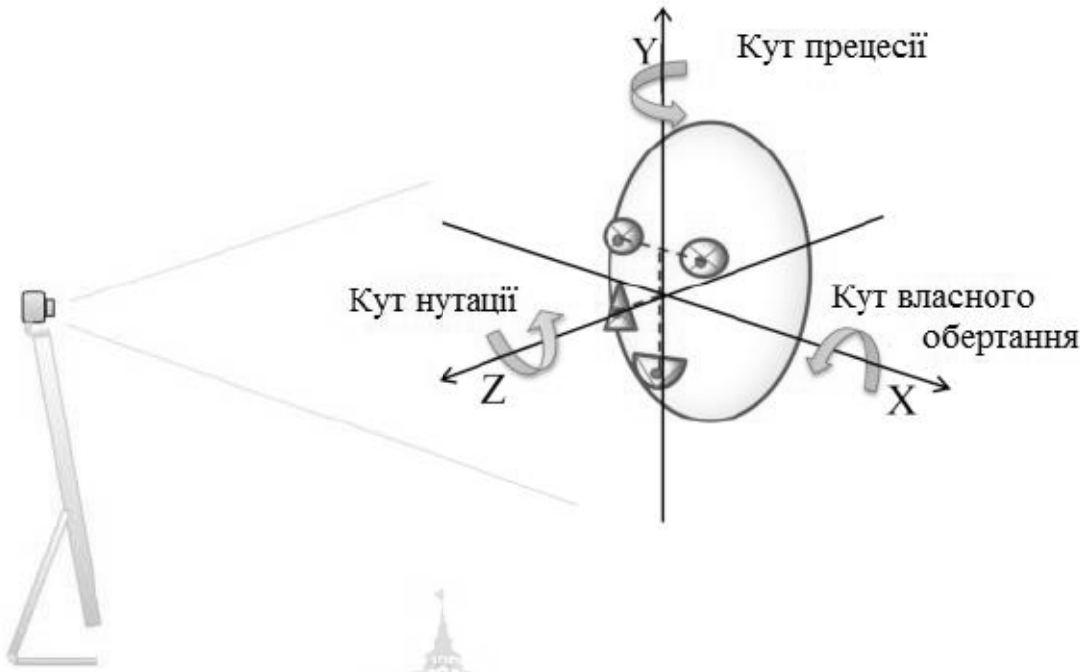

Рисунок 1.1 – Шість ступенів свободи, за якими можна обрахувати положення та напрям голови людини

Існують різноманітні активні дослідження у відстеженні положенні голови. Деякі системи покладаються на додаткові пристрої, що вдягаються на голову респондента та які дають зазвичай дуже точні результати. Недолік цих систем в тому, що вони потребують дороге додаткове обладнання. Саме тому будуть розглядатися методи пошуку положення голови, використовуючи тільки камеру, яка встановлена майже на будь-якому комп'ютері.

Ця частина задачі є складною, тому що необхідно підтримувати різні рівні освітленості, різні положення голови відносно камери, різні розмірі обличчя в залежності від відстані користувача до камери. В решіт решт, кожне обличчя має свої антропометричні особливості.

Для того щоб знайти шість параметрів положення голови людини та частини обличчя, що будуть використовуватися подалі, існують такі методи: FBM (за технікою VJA або TM), на основі нейронних мереж, ABM (за технікою ASM або AAM).

Наступним етапом є пошук особливих точок ока. Необхідно та достатньо мати інформацію про кути ока та положення зіниці на оці. FBM за технікою TM або Хафа перетворень – це група методів, що на теперішній час слугують вирішенням цієї задачі

Як тільки особливі точки були знайдені, то це дасть можливість, використовуючи дослідженні факти про людське око, визначити направлення фокусування погляду людини. Також він описує загальну модель ока, яка допоможе визначити направлення погляду, через радіус приблизної сфери очного яблука R (приблизної, тому що насправді око має форму приплюснутої сфери), зіниця знаходиться на фронтовій частині ока (див. рис. 1.2 а). Вона використовується для знаходження кута повороту відносно центру ока (див. рис. 1.2 б).

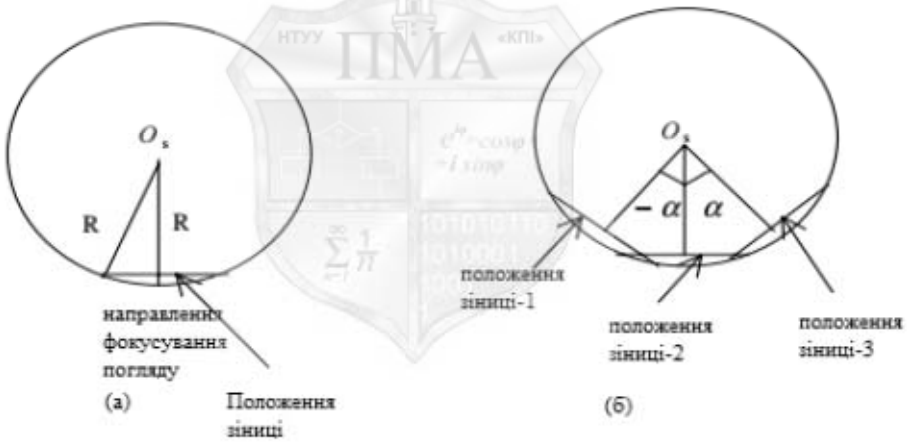

Рисунок 1.2 – Модель ока

Знаючи позицію очей в просторі та напрям фокусування погляду відносно кожного ока та знаючи положення моніторів відносно камери та їх розміри, можна, використовуючи звичайні геометричні перетворення та розв’язання рівняння з перетином прямої з площиною монітора з обмеженнями, визначити, на якій з моніторів дивиться користувач у даний момент чи дивиться він взагалі на монітор.

1.1 Аналіз предметної області

Як наукова дисципліна, комп'ютерний зір належить до теорії та технології створення штучних систем, які отримують інформацію у вигляді зображень. Відеодані можуть бути представлені у вигляді багатьох форм, таких як відеопослідовність, зображення з різних камер або тривимірними даними з медичного сканера, як зазначають Д. Форсейт та Ж. Понс у свої праці [9].

Наукова дисципліна комп'ютерного зору може бути охарактеризована як молода та різноманітна. І, хоча існують більш ранні роботи, можна сказати, що тільки з кінця 1970-х почалось інтенсивне вивчення цієї проблеми, коли комп'ютери змогли керувати обробкою великих наборів даних, таких як зображення. Однак, ці дослідження зазвичай починались з інших галузей, і, відповідно, нема стандартного формулювання проблеми комп'ютерного зору.

Також, і це навіть більш важливо, нема стандартного формулювання того, як повинна вирішуватись проблема комп'ютерного зору. Замість того, існує маса методів для вирішення різноманітних строго визначених задач комп'ютерного зору, де методи часто залежать від задач і рідко коли можуть бути узагальнені для широкого кола застосування.

Багато з методів та застосувань все ще знаходяться на стадії фундаментальних досліджень, але все більша кількість методів знаходить застосування в комерційних продуктах, де вони часто складають частину складнішої системи, яка може вирішувати складні задачі (наприклад, в галузі медичних зображень або вимірювання та контролю якості в процесах виробництва). В більшості практичних застосувань комп'ютерного зору комп'ютери попередньо запрограмовані для вирішення окремих задач, але методи, що базуються на знаннях, стають все більше узагальненими, стверджують у своїй роботі Люк'яниця та Шишкін [10].

Багато з досліджуваних питань можуть бути вивчені з суто математичної точки зору. Наприклад, більшість методів базуються на статистиці, оптимізаційній математиці або геометрії.

Нарешті, великі роботи ведуться в області практичного застосування комп'ютерного зору, в тому, як методи, що існують, можуть бути реалізовані програмно і апаратно чи як вони можуть бути змінені з метою досягнення високої швидкості роботи без істотного збільшення ресурсів, що споживаються.

1.2 Методи пошуку частин обличчя та вектору напряму голови людини

У цьому розділі розглядаються методи пошуку обличчя, а також методи пошуку напряму голови людини у просторі.

1.2.1 Feature Based Modelling Methods

Як визначає у своїй дисертації М. Сапенза [11], FBM оцінює положення голови за допомогою моделювання відношень між набором унікальних особливостей обличчя на зображенні, такі як очі, ніс та рот (див. рис. 1.3).

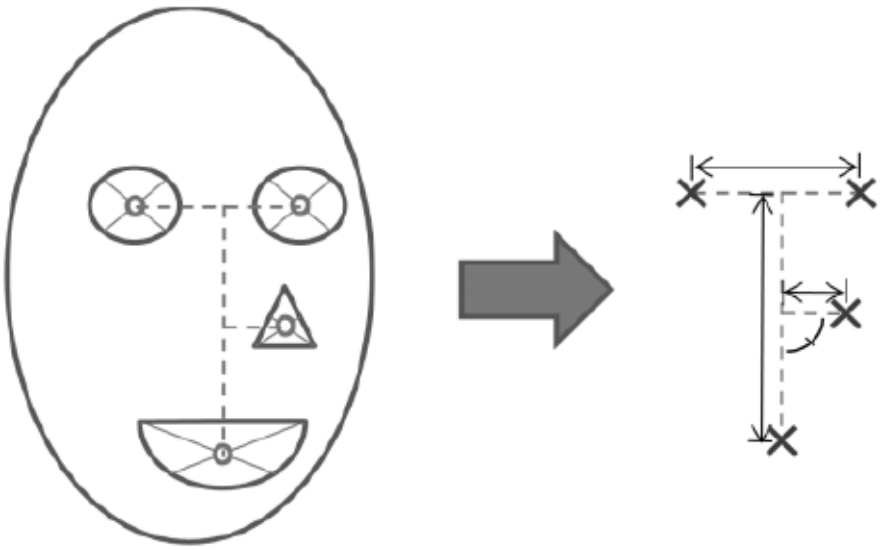

Рисунок 1.3 – Оцінка положення голови, базуючись на унікальних особливостях обличчя людини (ніс, рот, очі)

М. Сапенза у своїй праці [11] використовує наступний підхід: використовуючи статистичні дані о структурі обличчя разом з відомими відстанями між частинами обличчя та відомою конфігурацією камери (відома фокальна відстань камери), положення голови може бути оцінено. Відстань може бути знайдена за допомогою стеноп формули (1.1):

$$\frac{L_{obj}}{D} = \frac{L_{pixels}}{F} \tag{1.1}$$

де L_{obj} – це довжина об'єкту, L_{pixels} – довжина об'єкту на зображенні в пікселях, D – відстань від камери до об'єкту, F – фокальна відстань камери.

Інший спосіб знаходження позиції за відомими трьома точками на зображенні, якою займається проблема інверсивності та яку запропонував М. Фішер у своїй праці [12]. Для цього також потрібно знати статистичні або персональні (окремо для кожного користувача) характеристики обличчя. Як то відстань між очами, відстань між носом та ротом, тощо.

Дж. Драрені у своїй праці [13] запропонував простий спосіб знаходження положення голови, будуючи просту геометричну модель – циліндр, який базується на особливостях обличчя. Через його простоту дуже легко дізнатися

направлення та положення голови, але він потребує початкової оцінки положення особливостей обличчя. Це зобов'язує респондента фіксувати положення голови у відповідному напряму. І це потрібно робити при кожній помилці, що може виникнути під час пошуку положення голови.

1.2.2 FBM за технікою VJA

Як зазначено у роботах П. Віоли та М. Джонса [14] та [15], Основні принципи, на яких заснований метод, такі:

- використовуються зображення в інтегральному уявленні, що дозволяє обчислювати швидко необхідні об'єкти;
- використовуються ознаки Хаара, за допомогою яких відбувається пошук потрібного об'єкта (у даному контексті, обличчя та його рис);
- використовується бустінг (від англ. boost - поліпшення, посилення) для вибору найбільш підходящих ознак для шуканого об'єкта на даній частині зображення;
- всі ознаки надходять на вхід класифікатора, який дає результат «вірно» або «брехня»;
- використовуються каскади ознак для швидкого відкидання вікон, де не знайдено обличчя.

Для того, щоб проводити будь-які дії з даними, використовується інтегральне представлення зображень в методі Віоли-Джонса. Інтегральне представлення дозволяє швидко розраховувати сумарну яскравість довільного прямокутника на даному зображенні, причому якою б прямокутник не був, час розрахунку незмінно.

1.2.3 FBM за технікою TM

Р. Брунелі у своїй праці [16] стверджує, що основним принципом зіставлення шаблонів – є використання маски згортки (шаблон), з урахуванням конкретної функції пошуку зображення, який ми хочемо виявити.

Цей метод може бути легко виконаний на сірих зображень або так званих «крайових» зображеннях. Вихід згортки буде найвищим у місцях, де зображення структури відповідає структурі маски, де великі значення зображення стають помножені на більші значення маски.

1.2.4 Appearance Based Modelling Methods

М. Сапенза визначає у дисертації [11], що методи ABM представляють об'єкт як колекцію зображень з декількох точок зору. Положення голови оцінюється з використанням всієї виявленої області обличчя. Класифікація голови, як правило, здійснюється шляхом навчання моделі на безлічі помічених зображень з відомою позою. В випадку якщо положення голови невідомо, то використовуються деякі критерії подібності, щоб знайти найбільшу вірну відповідність.

1.2.5 ABM за технікою ASM

Ця техніка була розроблена Т. Кутесом і К. Тейлором в 1995 році [17]. Форми обмежені моделлю розподілу точок Статистичної форми моделі, щоб модель могла змінюватися тільки в межах розмічених прикладів з навчальної

вибірки. Форму об'єкта представляє безліч точок, контрольоване формою моделі. Мета алгоритму ASM – зіставити модель з новим зображенням. Алгоритм складається з двох чередуючих дій:

- Пошук на зображенні навколо кожної точки кращій позиції для даної точки.
- Оновлення параметрів моделі шляхом найкращої відповідності з новими знайденими позиціями

Щоб знайти кращу позицію для кожної точки можна шукати чіткі краї, а можна поєднати статистичну модель з тим, що очікується для даної точки. Оригінальний метод передбачає використання відстань Махаланобіса для обчислення кращій позиції для кожного орієнтира точки.

1.2.6 АВМ за технікою ААМ

Модель була вперше введена Г. Едвардсом, Т. Кутесом і К. Тейлором в контексті аналізу особи на 3-й Міжнародній конференції на розпізнавання жестів та обличчя, 1998 [18]. Т. Кутес, Г. Едвардс і К. Тейлор далі описали підхід, який широко використовується для зіставлення та відстеження особи та медичної інтерпретації зображень.

Алгоритм використовує різницю між поточною оцінкою зовнішнього вигляду та цільового зображення, збуджуючі процес оптимізації. Використовуючи метод найменших квадратів, модель може ставитись в відповідність у нових образів достатньо швидко.

1.2.7 Методи на основі нейронних мереж

Метод заснований на нейронних мережах був представлений Ж. Ліангом [19].

Дві нейронні мережі були натреновані для наближення функцій, що переводять зображення голови до орієнтації голови у просторі. Ілюстрацію етапу тренування мережі представлене на рисунку 1.4, де Xr, Yr, Zr – моменти обертання навколо OX, OY, OZ відповідно.

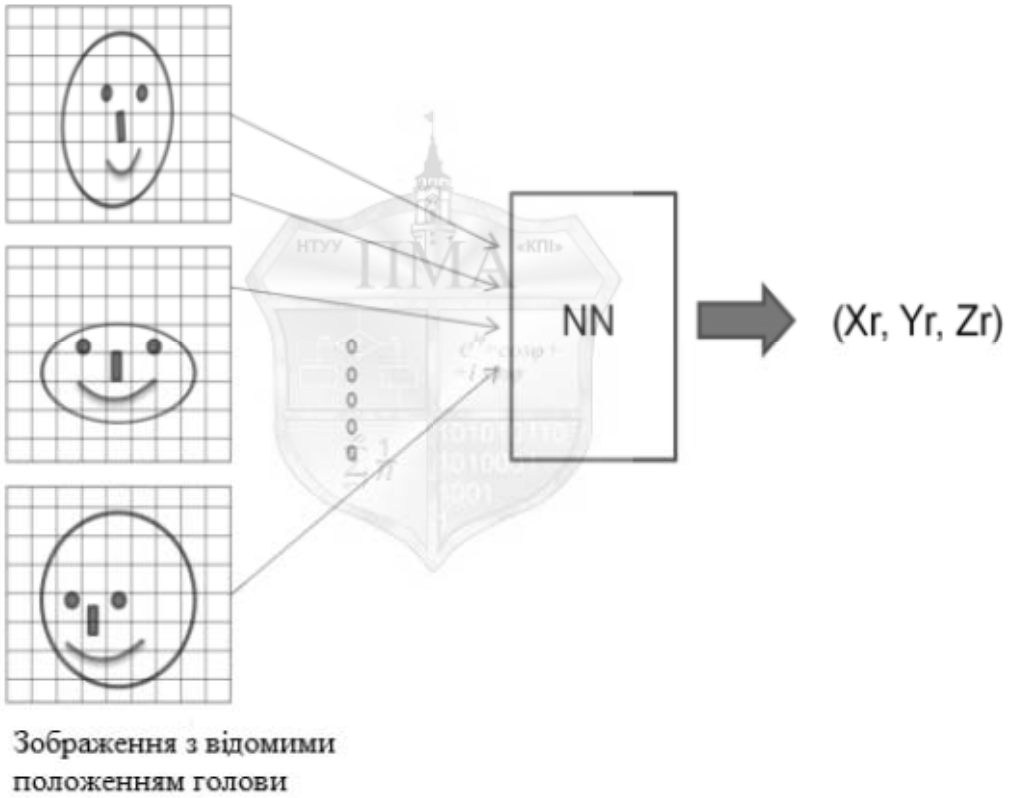

Рисунок 1.4. – Ілюстрація етапу тренування нейронної мережі, яка після тренування буде здатна класифікувати та оцінювати положення голови

Цей метод вимагає або великої кількості прикладів задачі розпізнавання (із правильними відповідями), або спеціальної структури нейронної мережі, що враховує специфіку даної задачі.

1.3 Методи пошуку особливих точок ока

У цьому розділі розглядаються існуючи методи пошуку особливих точок ока, що подалі дозволить знайти вектор фокусування погляду людини.

1.3.1 Feature Based Modelling Methods

Методи, що базуються на особливостях, оцінюють положення особливих точок ока, а саме лівий та правий кути ока (P_1 та P_2 відповідно) та зіниця (точка O). Ці особливі точки зображенні на рисунку 1.5 . Знаючи ці особливі точки, можна отримати радіус очного яблука.

Рисунок 1.5 – Модель ока з трьома особливими точками

Якщо попередній етап був зроблений за допомогою АВМ, то необхідність у пошуку кутів ока відпадає, так як активна форма обличчя, що формується, потребує точних границь ділянки ока. Знаючи цю активну форму, можна легко дізнатися і дві необхідні ключові точки. Залишається проблема пошуку самої зіниці.

У випадку виконання попереднього етапу методом нейронних мереж, то положення очей на обличчі все ж остається невідомим. Це забов'язує робити додатковий крок за допомогою виконання інших методів, як АВМ або FBM, для пошуку області інтересу очей.

При оцінюванні положення голови за допомогою FBM потрібно знайти усі три особливі точки, але область інтересу вже відома відмінно від методу, що базується на нейронних мережах.

1.3.2 FBM за технікою ТМ

Цей метод вже був описаний в розділі 1.2.3.

Для пошуку особливих точок він базується на тих ж самих принципах, що й в 1.2.3. Додатковою модифікацію є додавання якірних точок, що позначають розташування особливої точки на шаблоні.

При знаходженні згортки на початковому зображенні дозволить перенести якірну точку в область інтересу на зображенні.

1.3.3 FBM за технікою Хафа перетворень

Р. Дуда описує у свої роботі один за найпопулярніших методів обробки зображень [20].При автоматизованому аналізі цифрових зображень дуже часто виникає проблема ідентифікації простих фігур, таких як прямі, кола або еліпси. У багатьох випадках використовується алгоритм пошуку кордонів як передобробки для отримання точок, що знаходяться на кривій в зображенні.

Однак, або через зашумленості зображення, або через недосконалість алгоритму виявлення кордонів, можуть з'явитися «втрачені» точки на кривій, також як і невеликі відхилення від ідеальної форми прямої, кола чи еліпса. З цих причин часто досить складно приписати знайдені кордону відповідною прямим, колам і еліпсах в зображенні.

Призначення перетворення Хафа – розв'язати проблему угруповання граничних точок шляхом застосування певної процедури голосування до набору параметрезованих об'єктів зображення.

Перетворення Хафа ефективно тільки при значній кількості «попадань» в відповідний елемент простору Хафа, тільки тоді можна з упевненістю визначити фігуру, нехтуючи фоновим шумом. Це означає, що розмір елемента не повинен бути дуже маленьким, інакше деякі значення потраплять в сусідні елементи, зменшуючи видимість потрібного елемента.

Також, коли число параметрів велике (більше трьох), середня кількість «попадань» в елемент невелика, і тому вірний елемент не буде дуже сильно відрізнятися від сусідів. Таким чином, алгоритм повинен використовуватися з великою обережністю, щоб не визначити щось інше як прямі і кола.

1.4 Порівняння методів

Виходячи з постановки задачі, а також вимог до самого програного засобу, були обрані наступні критерії для вибору найоптимальнішого методу:

- Простота методу. Під цим розуміється, що метод, який буде реалізовуватися, повинен мати короткий підготовчий етап для вдалого виконання програми, як то тренування вибірки для методу, збір персональних даних про користувача, тощо.
- Швидкість виконання методу. Під цим розуміється, що час виконання у зв’язці з іншими методами не повинен бути більше ніж 100 мс на одне зображення.
- Стійкість виконання методу. Під цим розуміється, що метод має працювати при різних рівнях та напрямах освітленості, кількості шуму, положення об’єкту.

- Точність виконання методу. Під цим розуміється, що кількість «правильних» виконань методу (правильно знайшлась особлива точка, правильно знайшлась ділянка інтересу, тощо) має перевищувати або дорівнювати 90 відсоткам.
- Ресурси необхідні для виконання методу.

Пріоритет критеріїв при пошуку найоптимальнішого методу йде з більшого до меншого за порядком, в якому вони були описані вище.

В таблиці 1.1 порівнюються методи, що необхідні для пошуку частин обличчя та напряму голови людини на зображенні. В той час, як в таблиці 1.2 порівнюються методи, що необхідні для пошуку особливих точок в області очей на обличчі людини.

Велика увага приділяється до методів, що потребують навчання для подальшого виконання поставленої задачі перед методом.

Такі навчання зазвичай потребують великої кількості навчальних зображень з позитивними та негативними результатами, що ускладнює роботу з ними.

Таблиця 1.1 – Порівняльна таблиця методів пошуку частин обличчя та напряму голови людини на зображенні

	Простота	Швидкість	Стійкість	Точність	Ресурси
FBM (TM)	Необхідна наявність шаблону у вигляді зображень	~10 мс	Стійкий до перепадів освітленості, невеликих поворотів об'єкту	+	Модифіковане інтегральне зображення, місце для шаблонів
FBM (VJA)	Необхідно натренувати каскади Хаара для роботи	~33 мс	+	+	Інтегральне зображення
Нейронні мережі	Необхідно натренувати мережу для роботи, велика вибірка для тренування	~ 5мс	Залежить від тренувальної вибірки	+	Не займає додаткових ресурсів
ABM (AAM)	Необхідно натренувати модель для роботи	~52 мс	Залежить від тренувальної вибірки	+	Не займає додаткових ресурсів
ABM (ASM)	Необхідно натренувати модель для роботи	~45 мс	Залежить від тренувальної вибірки	+	Місце для шаблонів локальних точок обличчя

Таблиця 1.2 – Порівняльна таблиця методів пошуку особливих точок в області очей людини

	Простота	Швидкість	Стійкість	Точність	Ресурси
FBM (TM)	Необхідна наявність шаблону у вигляді зображень	~10 мс	+	+	Модифіковане інтегральне зображення, місце для шаблонів
FBM (перетворення Хафа)	+	~5 мс	Дуже сильно залежить від рівня освітлення та шуму	+	Додаткова матриця для збереження результати згортки

Висновки до розділу 1

Розглядаючи математичні методи для знаходження частин обличчя людини, положення голови людини та визначення особливих точок в області очей можна виділити такі основні підходи: FBM (за технікою VJA, TM або Хафа перетворень), на основі нейронних мереж, ABM (за технікою ASM або AAM).

Для того щоб застосовувати FBM за VJA, нейронні мережі, ABM потрібно навчати систему для роботи з зображеннями (навчання мережі, навчання каскадів, тощо), що часто є важким завданням. Також FBM за технікою Хафа перетворень не є точним для описання особливих точок людини, що пов'язано з тим, що існують багато точок, що схожі за характеристиками з необхідними точками.

2 ВИБІР МЕТОДУ РІШЕННЯ

Виходячи з попереднього розділу було обрано метод шаблонів – як основний метод пошук частин обличчя, а також особливих точок ока.

Також, дивлячись на інші методи, що не були до реалізації у програмному засобі, було вирішено модифікувати базовий метод шаблонів за допомогою модифікованого інтегрального зображення.

Ця модифікація дозволить прискорити метод, виключаючи постійне обчислення квадратів для нормалізації методу.

Для відстеження фокусування погляду людини також необхідно знати його положення голови. Для цього буде використовуватися інверсія перспективи для трьох точок. Додаткова буде аналізуватися четверта точка, що дозволить виділити необхідні точки із можливих варіантів.

2.1 Інтегральне представлення зображення

Інтегральне представлення зображення – це матриця, розмірність якої збігається з розмірністю вихідного зображення, вперше була запропонована П. Віолой та М. Джонсом у своїх працях [14][15]. Елементи цієї матриці розраховуються за формулою (2.1):

$$I(x,y) = \sum_{x' \le x, y' \le y} i(x',y') \tag{2.1}$$

де i(x’, y’) - яскравість пікселя вихідного зображення.

Кожен елемент інтегрального зображення I[x, y] містить в собі суму пікселів зображення в прямокутнику від (0,0) до (x, y).

Розрахунок інтегрального зображення займає лінійний час, пропорційне числу пікселів вихідного зображенні.

Розрахунок інтегрального зображення можна призводить за рекурентною формулою (2.2):

$$I(x,y) = i(x,y) - I(x-1,y-1) + I(x,y-1) + I(x-1,y) \qquad (2.2)$$

Однією з корисних особливостей інтегрального подання є можливість дуже швидко вирахувати суму пікселів довільного прямокутника (або будь-який інший фігури, яку можна апроксимувати декількома прямокутниками). Наприклад, нас цікавить прямокутник ABCD (рис. 2.1).

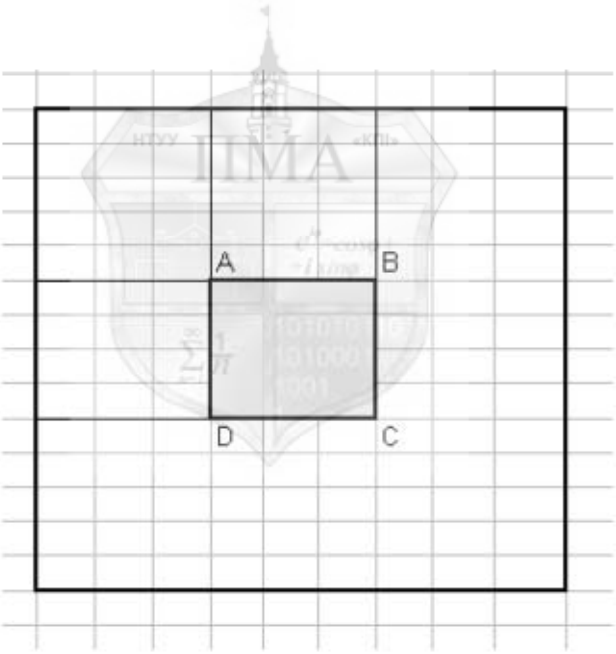

Рисунок 2.1 – Прямокутник, суму пікселів якої треба обрахувати

Суму усередині прямокутника можна виразити через суми і різниці суміжних прямокутників за формулою (2.3)(див. рисунок 2.2 для прикладу) [13]:

$$Sum(ABCD) = I(A) + I(C) - I(B) - I(D) \qquad (2.3)$$

32	39	2	20	8	13	2	5	3
15	14	11	12	13	14	5	20	8
7	14	1	0	14	6	5	13	8
5	13	10	13	8	1	12	20	1
0	2	12	20	15	7	19	30	48
23	54	11	10	76	30	44	11	22
11	25	77	22	15	42	23	5	12
45	33	65	3	17	8	90	6	48
25	1	54	54	88	6	13	56	23
13	8	9	23	0	78	4	42	11
28	98	12	45	18	34	7	28	75
79	14	6	43	33	89	14	41	28
21	32	14	13	6	69	68	60	26
34	76	98	24	8	5	17	18	13

10	76	30				**S =**	**371**
22	15	42					
3	17	8					
54	88	6			**A+C-B-D =**		**371**

32	39	2				**A**	177
15	14	11					
7	14	1					
5	13	10					
0	2	12					
32	39	2	20	8	13	**B**	341
15	14	11	12	13	14		
7	14	1	0	14	6		
5	13	10	13	8	1		
0	2	12	20	15	7		
32	39	2	20	8	13	**C**	1136
15	14	11	12	13	14		
7	14	1	0	14	6		
5	13	10	13	8	1		
0	2	12	20	15	7		
23	54	11	10	76	30		
11	25	77	22	15	42		
45	33	65	3	17	8		
25	1	54	54	88	6		
32	39	2				**D**	601
15	14	11					
7	14	1					
5	13	10					
0	2	12					
23	54	11					
11	25	77					
45	33	65					
25	1	54					

Рисунок 2.2 – Приклад роботи алгоритму в середовищі Excel

2.2 Модифікація інтегрального зображення

Ідея модифікації інтегрального зображення складається в зберіганні суми квадратів яскравості зображення. Тобто в кожній клітинці зображення буде сума квадратів яскравості (формула 2.4).

$$I(x,y) = \sum_{x' \le x, y' \le y} i(x',y')^2 \tag{2.4}$$

Усі операції, що були дійсні для звичайного інтегрального представлення зображення, є дійсні в модифікованому варіанті. Різниця лише в тому, що результатом цих операцій є сума квадратів яскравості зображення.

2.3 Метод зіставлення шаблонів

Основний метод зіставлення шаблонів використовує маску згортки (шаблон) з урахуванням конкретної функції пошуку зображення, який ми хочемо виявити.

Цей метод може бути легко виконаний на сірих зображень або так званих «крайових» зображеннях. Вихід згортки буде найвищим у місцях, де зображення структури відповідає структурі маски, де великі значення зображення стають помножені на великі значення маски.

Цей метод, як правило, здійснюється, в першу чергу, вибираючи частину зображення для пошуку. Назвемо пошук зображення S(X,Y), де (X,Y) представляють координати кожного пікселя в пошуковій частині зображення.

Будемо називати шаблон T (x,y), де (x, y) представляють координати кожного пікселя в матриці інтегрального зображення. Тоді просто за допомогою перенесення центру (або походження) форми T(x, y) над кожною (X, Y) точки в пошуковій частині зображення і розрахунку суми виконання функції між коефіцієнтами в S(X, Y) і T(x, y) по всій площі, натягнутої на шаблон. Як і всі можливі позиції шаблону по відношенню до пошукової частини зображення, вважається, позиція з найвищою (в залежності від функції порівняння) кількістю очок – є краща позиція. Цей метод іноді називають "лінійна просторова фільтрація" і шаблон називається маска-фільтр.

Піксель в пошуковій зображення з координатами (X,Y) має інтенсивність (X,Y) і пікселів в шаблоні з координатами (x, y) має інтенсивність (x, y). Таким чином, абсолютна відмінність в інтенсивностях пікселів визначається як зазначено в нормалізованій кореляційній формулі 2.5.

$$S(x,y) = \frac{\sum_{i=0,j=0}^{T_{rows},T_{cols}} \left(T(i,j) \cdot I(x+i,y+j)\right)}{\sqrt{\sum_{i=0,j=0}^{T_{rows},T_{cols}} T(i,j)^2 \cdot \sum_{i=0,j=0}^{T_{rows},T_{cols}} I(x+i,y+j)^2}} \quad (2.5)$$

Чим більше значення функції в точці, тим більш вірогідно, що ця область є збігом початкового зображення зі шаблоном.

2.4 Модифікація методу зіставлення шаблонів

Надихнувшись ідеєю оптимізації методу Віоли-Джонса, було вирішено за схожим принципом модифікувати, та таким чином прискорити, метод зіставлення шаблонів.

Якщо подивитись на нижню частину формули 2.5, то можна зауважити, що кожен раз обраховується сума квадратів інтенсивності шаблону та скануючого вікна початкового зображення.

Ці обрахунки можна звести до часу лінійного виконання за допомогою описаного раніше модифікованого інтегрального представлення зображення, що зберігає суми квадратів інтенсивностей зображення, та застосувати до них операції знаходження суми, що описані в підрозділі 2.1 цього розділу.

2.5 Інверсія перспективи

Ми припускаємо, що ми визначили групу точок на зображенні, що з'єднання між собою прямими та представляють реальний об’єкт з відомими параметрами у 3D-просторі.

Тепер ми повинні вирішити проблему з перспективи з точки зображення до точки, що описує реальний об'єкт. Тобто дається набір з 3 точок, при якому є більше восьми можливих рішень, з яких тільки чотири відбуваються в реальному 3D-просторі в передній частині камери. Якщо є додаткова четверта точка і ці точки не лежать в одній площині, є два рішення. Але із-за особливості положення людини перед камерою це рішення залишається унікальним, тобто єдиним.

Ми будемо мати справу тільки з випадку 3 пунктів, показаних на рисунку 2.3.

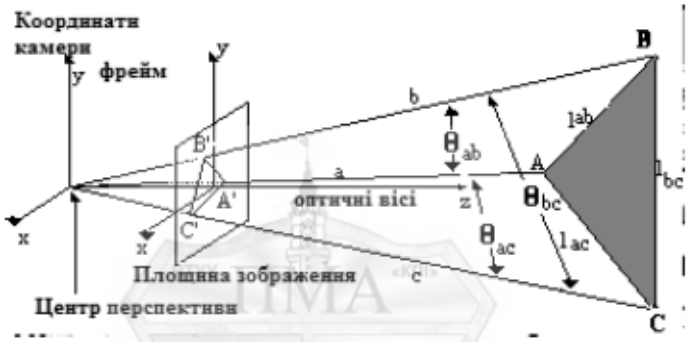

Рисунок 2.3 – Ілюстрація проблеми інверсії перспективи за трьома точками на зображенні з позначеннями оптичних прямих, та точок, що відносяться до реального об'єкта та його проекції на площині зображення

Основна проблема полягає у визначенні довжини трьох так званих «ногах» тетраедра, утвореного A, B, C і в центрі перспективи. З тригонометрії, ми можемо сформувати три рівняння (2.6, 2.7, 2.8), заснованого на верховенстві косинуса,

$$l_{ab}^2 = a^2 + b^2 - 2ab \cdot \cos\theta_{ab} \quad (2.6)$$

$$l_{ac}^2 = a^2 + b^2 - 2ac \cdot \cos\theta_{ac} \quad (2.7)$$

$$l_{bc}^2 = b^2 + c^2 - 2bc \cdot cos\theta_{bc} \quad (2.8)$$

Виконання заміни (2.9).

$$b = k_1 a \quad c = k_2 a \tag{2.9}$$

Проводячи заміну відстаней до точки B і C в якості відстані до точки A, отримуємо наступні формули (2.10, 2.11, 2.12).

$$l_{ab}^2 = a^2 + k_1^2 a^2 - 2a^2 k_1 \cdot \cos\theta_{ab} \tag{2.10}$$

$$l_{ac}^2 = a^2 + k_2^2 a^2 - 2a^2 k_1 \cdot \cos\theta_{ac} \tag{2.11}$$

$$l_{bc}^2 = k_1^2 a^2 + k_2^2 a^2 - 2a^2 k_1 k_2 \cdot cos\theta_{bc} \tag{2.12}$$

Шляхом комбінування і перебудови рівнянь 2.10,2.11 і 2.12, щоб утворити рівняння четвертого степеня (2.13).

$$q_4 k_1^4 + q_3 k_1^3 + q_2 k_1^2 + q_1 k_1 + q_0 = 0 \tag{2.13}$$

Де

$$q_4 = (K_1K_2 - K_1 - K_2)^2 - 4K_1K_2(\cos\theta_{bc})^2$$

$$q_3 = 4(K_1K_2 - K_1 - K_2)K_2(1 - K_1)\cos\theta_{ab} + 4K_1\cos\theta_{bc}\left((K_1K_2 + K_2 - K_1)\cos\theta_{ac}\right)$$

$$q_2 = (2K_2(1 - K_1)\cos\theta_{ab})^2 + 2(K_1K_2 + K_1 - K_2)(K_1K_2 - K_1 - K_2) + 4K_1\big((K_1 - K_2)(\cos\theta_{bc})^2 + (1 - K_2)K_1(\cos\theta_{ac})^2 - 2K_2(1 + K_1)\cos\theta_{ab}\cos\theta_{ac}\cos\theta_{bc}\big)$$

$$q_1 = 4(K_1K_2 + K_1 - K_2)K_2(1 - K_1)\cos\theta_{ab} + 4K_1\big((K_1K_2 - K_1 + K_2)\cos\theta_{ac}\cos\theta_{bc} + 2K_1K_2\cos\theta_{ab}(\cos\theta_{ac})^2\big)$$

$$q_0 = (K_1K_2 + K_1 - K_2)^2 - 4K_1^2K_2(\cos\theta_{ac})^2$$

та

$$K_1 = \frac{l_{bc}^2}{l_{ac}^2} \qquad K_2 = \frac{l_{bc}^2}{l_{ab}^2}$$

l_{ab}, l_{ac}, l_{bc} – відомі параметри з реальної моделі об'єкта. Для кожного позитивного дійсного кореня рівняння 2.13 ми можемо обчислити значення, B і C, тобто замінюючи k1 в рівнянні 2.10 нижче в рівнянні 2.14 та 2.15. Щоб визначити c, ми також повинні обчислити рівняння 2.16. Є два можливих умови, наведені в рівняннях 2.17 і 2.18.

$$a = \frac{l_{ab}}{\sqrt{k_1^2 - 2k_1 \cos\theta_{ab} + 1}} \tag{2.14}$$

$$b = ak_1 \tag{2.15}$$

$$c = ak_2 \tag{2.16}$$

де, або

$$k_2 = \frac{k_1 \cos\theta_{bc}\,(k_1^2 - K_1) - 2(K_1 \cos\theta_{ac} - k_1 \cos\theta_{bc})}{(1 - K_1)\big(k_1^2(1 - K_2) + 2k_1K_2 \cos\theta_{ab} - K_2 - (k_1^2 - K_1)\big)} \tag{2.17}$$

або якщо подільник дорівнює нулю

$$k_2 = \cos\theta_{ac} \pm \sqrt{\cos\theta_{ac}{}^2 + \frac{l_{ac}^2 - a^2}{a^2}} \tag{2.18}$$

Висновки до розділу 2

Були розглянуті математичні методи, що будуть використовуватися у дипломній роботі.

Вирішено використовувати метод зіставлення шаблонів, як основний метод пошуку особливих точок та частин обличчя.

Було модифіковано інтегральне представлення зображення, що зберігає суму квадратів інтенсивності кожного пікселя.

Для зменшення часу обрахунку проміжних даних, метод зіставлення шаблонів був модифікований за допомогою модифікованого інтегрального представлення зображення.

3 ПОБУДОВА МАТЕМАТИЧНОЇ МОДЕЛІ

Так як програма працює з відеопотоком та зображеннями, ми розберемо ці поняття з математичної точки зору. Не обмежуючи спільності, будемо вважати, що математична модель зображення (МІ) задана на багатовимірних прямокутних сітках з одиничним кроком. На рис. 3.2, а і 3.2, б зображені двовимірна і тривимірна сітки.

У загальному випадку зображення (I) задано у вузлах n-мірної сітки (формула 3.1), в кожній клітинці якої знаходиться вектор цілочисельних значень (формула 3.2).

$$I \equiv \begin{pmatrix} i_{11} & \cdots & i_{1n} \\ \vdots & \ddots & \vdots \\ i_{m1} & \cdots & i_{mn} \end{pmatrix}, i_{kl} \in C \tag{3.1}$$

$$\bar{c} \in C, \qquad \bar{c} \equiv (r \quad g \quad b), \quad r, g, b \in \{N, 0\} \tag{3.2}$$

Кожен вектор має довжину 3.Ці значення позначають описуються за допомогою моделі RGB.

RGB (скорочено від англ. Red, Green, Blue — червоний, зелений, синій) — адитивна колірна модель, що описує спосіб синтезу кольору, за якою червоне, зелене та синє світло накладаються разом, змішуючись у різноманітні кольори (рис. 3.1).

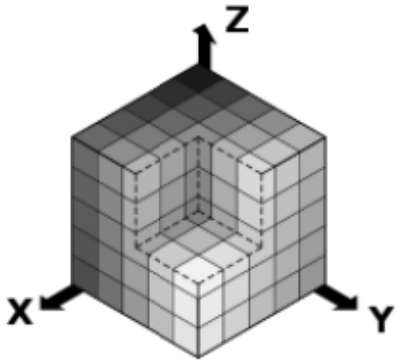

Рисунок 3.1 – Тривимірна модель RGB

Якщо дані представляють собою тимчасову послідовність I, то іноді зручно вважати цю послідовність одним I, збільшивши розмірність сітки на одиницю. Наприклад, послідовність з плоских I (рис. 3.1, а) можна розглядати як одне тривимірне I (рис. 3.1, б).

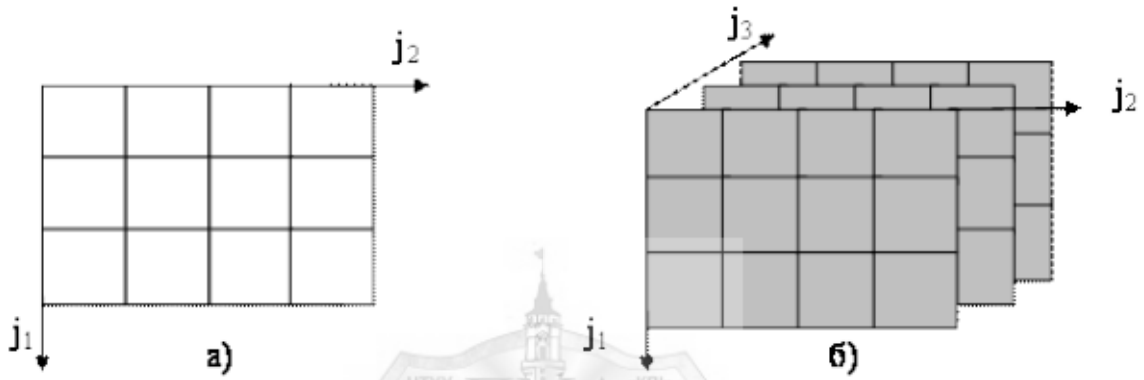

Рисунок 3.2 – двовимірне (а) та трьохвимірне (б) I (воно ж MI)

Значення елементів I неможливо точно передбачити заздалегідь (інакше система спостереження була б не потрібна), тому природно розглядати ці значення як випадкові величини (ВВ).

Для переводу трьох-канального зображення до одно-канального зображення використовується лінійне перетворення за формулою 3.3.

$$I(x,y) = 0.299 \cdot R(x,y) + 0.587 \cdot G(x,y) + 0.114 \cdot B(x,y) \qquad (3.3)$$

де I – це одно-канальне зображення, де значення позначають інтенсивність самого зображення, R, G, B – це красний, зелений та синій канали відповідно вихідного зображення.

Для того, щоб робити які-небудь дії з даними, використовується інтегральне представлення зображень взятий в методі Віоли-Джонса.

Інтегральне представлення зображення – це матриця, що збігається за розмірами з вихідним зображенням. У кожному елементі її зберігається сума інтенсивностей всіх пікселів, що знаходяться лівіше і вище даного елемента. Елементи матриці розраховуються за формулою 3.4.

$$L(x,y) = \sum_{i=1,j=1}^{i\leq x, i\leq y} I(i,j)^2 \qquad (3.4)$$

де $I(i,j)$ – яскравість пікселя вихідного зображення.

Розрахунок матриці можливий за формулою 3.5.

$$L(x,y) = I(x,y)^2 - L(x-1,y-1) + L(x,y-1) + L(x-1,y) \qquad (3.5)$$

Ознака – відображення f: X => D_f, де D_f - безліч допустимих значень ознаки. Якщо задані ознаки f_1, ..., f_n, то вектор ознак x = (f_1 (x), ..., f_n (x)) називається ознаковим описом об'єкта x ∈ X. Ознаковому опису допустимо ототожнювати з самими об'єктами. При цьому безліч X = D_{f1} * ... * D_{fn} називають просторі ознак.

Обчислюваним значенням такої ознаки приведене за формулою 3.6.

$$F = X - Y \qquad (3.6)$$

де X - сума значень яскравостей точок, що закриваються світлою частиною ознаки, а Y - сума значень яскравостей точок, що закриваються темною частиною ознаки. Для їх обчислення використовується поняття інтегрального зображення, розглянуте вище.

Позначимо T як деякий шаблон точки або об’єкту, що нам необхідно знайти. T представляє у вигляді деякого зображення.

Для знаходження ділянки на початковому зображенні за шаблоном мінімізується функція S, яка описується формулою 3.7.

$$S(x,y) = \frac{\sum_{i=0,j=0}^{T_{rows},T_{cols}} \left(T(i,j) \cdot I(x+i,y+j)\right)}{\sqrt{\sum_{i=0,j=0}^{T_{rows},T_{cols}} T(i,j)^2 \cdot \sum_{i=0,j=0}^{T_{rows},T_{cols}} I(x+i,y+j)^2}} \quad (3.7)$$

Завдяки пошуку за шаблоном вдається знайти такі ділянки як рот, ніс, очі, зіниця, кути ока, тощо. Всі ці ділянки представлені у вигляди областей інтересу, що були зазначені вище.

Кожна область інтересу або містить якірну точку, що описана для кожної області окремо, або містить центральну точку, яка формується за формулою 3.8.

$$\mathrm{C} = \frac{A+B}{2} \quad (3.8)$$

де A і B – це точки, що описують область інтересу у вигляді прямокутника. Ці центральні точки подалі використовуються для знаходження необхідний моделей в трьохвимірному декартовому просторі, де початком координат вважається камера.

Як тільки усі області інтересу були знайдені, на поточному зображені формується модель голови людини, яка необхідна лише для знаходження відносного положення. Модель голови людини виражається через точки у трьохвимірному просторі (див. рис. 3.3):

$$M = \left(m_x, m_y, m_z\right)$$

$$LE = \left(le_x, le_y, le_z\right)$$

$$RE = \left(re_x, re_y, re_z\right)$$

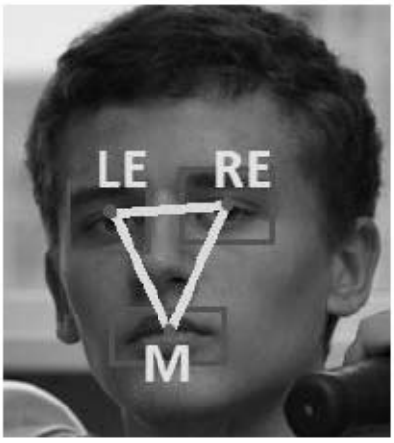

Рисунок 3.3 – Модель голови людини у вигляді трикутника, що складається з трьох особливих точок

Далі формується модель монітору. Ця модель виражається через послідовність точок у трьохвимірному просторі (див. рис. 3.4):

$$MC = (mc_x, mc_y, mc_z)$$

$$UC = (uc_x, uc_y, uc_z)$$

$$DC = (dc_x, dc_y, dc_z)$$

$$LC = (lc_x, lc_y, lc_z)$$

$$RC = (rc_x, rc_y, rc_z)$$

$$UR = (ur_x, ur_y, ur_z)$$

$$UL = (ul_x, ul_y, ul_z)$$

$$DR = (dr_x, dr_y, dr_z)$$

$$DL = (dl_x, dl_y, dl_z)$$

Рисунок 3.4 – Модель монітору, що виражається через послідовність особливих точок

За трьома точками моделі обличчя людини будується площина (3.9), до якої шукається нормаль (3.10). Ця нормаль позначає напрям голови людини.

$$\begin{vmatrix} x - m_x & y - m_y & z - m_z \\ re_x - m_x & re_y - m_y & re_z - m_z \\ le_x - m_x & le_y - m_y & le_z - m_z \end{vmatrix} = C_1 x + C_2 y + C_3 z + C_4 = 0 \qquad (3.9)$$

$$\bar{n} = (C_1, C_2, C_3) \qquad (3.10)$$

Як тільки знайшли вектор напряму голови людини, будується модель кожного ока людини. Вона складається з послідовності чотирьох точок (див. рис. 3.5):

$$EC = (ec_x, ec_y, ec_z)$$

$$UC = (pc_x, pc_y, pc_z)$$

$$ERE = (ere_x, ere_y, ere_z)$$

$$ELE = (ele_x, ele_y, ele_z)$$

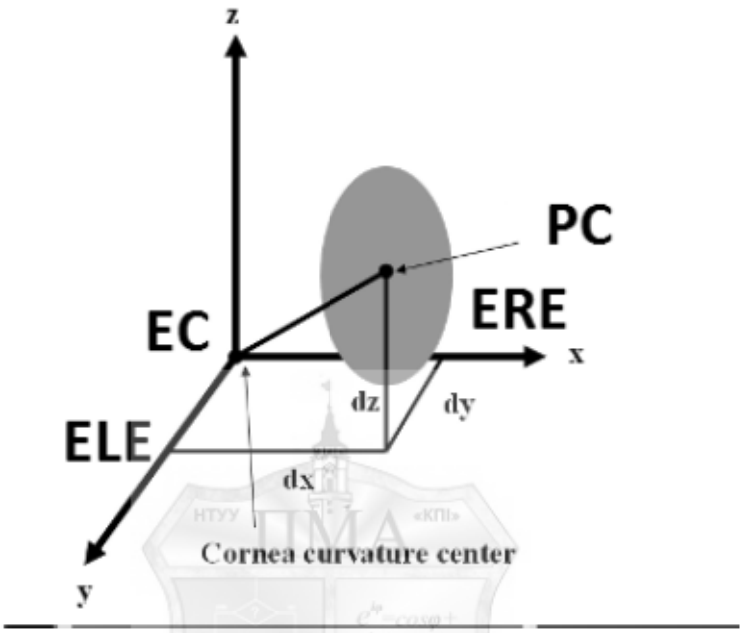

Рисунок 3.5 – Модель ока людини, що складається з чотирьох особливих точок

Використовуючи кут (3.11), що утворюється використовуючи модель ока, будується вектор погляду людини (3.12):

$$\phi = \arccos\left(\frac{\left(\overline{(EC, PC)}, \overline{(EC, ELE)}\right)}{\left|\overline{(EC, PC)}\right| \cdot \left|\overline{(EC, ELE)}\right|}\right) - \pi/4 \qquad (3.11)$$

$$\overline{gaze} = \cos(\phi) \ \cdot \bar{n} \qquad (3.12)$$

Знайшовши вектор погляду, можливо легко знайти з яким монітором він перетинається, побудувавши площину кожної моделі монітора за її точками.

Нам відомо, що якщо точка належить деякій прямій, то координати точки задовольняють рівнянням прямої. Аналогічно, якщо точка лежить в деякій площині, то координати точки задовольняють рівняння цієї площини. За визначенням точка перетину прямої і площини є загальною точкою прямої і

площини, тоді координати точки перетину задовольняють як рівнянням прямої, так і рівняння площині.

Таким чином, для вирішення поставленого завдання нам слід підставити координати точки FOCUS в задані рівняння прямий і в рівняння площини. Якщо при цьому всі рівняння звернуться до вірні рівності, то точка FOCUS є точкою перетину заданих прямої і площини, в іншому випадку точка FOCUS не є точкою перетину прямої і площини.

Далі перевіряється, щоб точка FOCUS не виходила за межі монітора. Для цього виконується наступні перевірки (3.13-3.14):

$$lc_x < focus_x < rc_x \tag{3.13}$$

$$dc_y < focus_y < uc_y \tag{3.14}$$

Якщо ця умова виконується, то користувач фокусується на цьому моніторі.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ

Як варіанти розглядались такі мови програмування: Qt/QML, C, C++, Python, Pascal, Java.

В якості середовища розглядались такі варіанти: Qt Creator, Visual Studio, Eclipse, Flash Develop.

Для розробки програмного засобу було вирішено обрати Visual Studio 2013 в якості середовища програмування. Зберігання даних та деякі функції були використані за допомогою популярної бібліотеки комп'ютерного зору с відкритим вихідним кодом OpenCV. Вся програма в основному була написана на мові програмування C++.

Для полегшення розуміння роботи програма також була описана її структура та алгоритм роботи. Це дозволить подалі виявити можливі недоліки та поліпшити роботу самої програми.

Для взаємодії з самою програмою далі наведено керівництво користувача. Це допоможе розібратись з існуючими опціями користуванням програмою та способами виведення необхідної користувачу даними.

В кінці розділу наведене тестування різних компонентів програми. Описані можливі похибки та їх причини.

4.1 Вибір технологій розробки

У даному розділі описується обрані технології розробки та їх переваги, які будуть застосовуватися в програмному засобі.

4.1.1 Мова програмування

Виходячи з роботи Строуструпа [21], при створенні C++ прагнули зберегти сумісність з мовою C. Більшість програм на C справно працюватимуть і з компілятором C++. C++ має синтаксис, заснований на синтаксисі C (див. список операторів мов C та C++).

Нововведеннями C++ порівняно з C є:

- об'єктно-орієнтованого програмування через класи;
- підтримка узагальненого програмування через шаблони;
- доповнення до стандартної бібліотеки;
- додаткові типи даних;
- обробка винятків;
- простори імен;
- вбудовані функції;
- перевантаження операторів;
- перевантаження імен функцій;
- посилання і оператори управління вільно розподіленою пам'яттю.

У 1998 році ратифіковано міжнародний стандарт мови C++: ISO/IEC 14882 «Standard for the C++ Programming Language». Поточна версія цього стандарту — ISO/IEC 14882:2011.

4.1.2 Середовище розробки

Microsoft® Visual Studio® 2013 - найпоширеніша в нашій країні середовище розробки ПЗ. Сьогодні це основне і найефективніший засіб розробки рішень для платформи Microsoft, як це описано на офіційному сайті

розробників цієї платформи [22]. Також доступна безкоштовна версія для студентів.

Visual Studio 2013 уособлює собою уявлення корпорації Майкрософт про інтелектуальні клієнтських додатках і дозволяє швидко створювати підключаються до баз даних програми, здатні забезпечити найширші можливості для роботи користувачів. За допомогою Visual Studio 2013 можна збирати та аналізувати інформацію простіше, ніж коли б то не було раніше, що сприяє прийняттю ефективних бізнес-рішень.

За допомогою Visual Studio 2013 можливо швидко створювати більш безпечні, керовані і надійні додатки, що використовують переваги Windows 8™ і системи Microsoft Office 2013.

4.1.3 Основна бібліотека

В якості основної бібліотеки для вирішення задачі було обрано OpenCV.

Як позиціонують розробники на своєму сайті [23], OpenCV (англ. Open Source Computer Vision Library, бібліотека комп'ютерного зору з відкритим вихідним кодом) — бібліотека функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом.

Бібліотека містить понад 2500 оптимізованих алгоритмів, серед яких повний набір як класичних так і практичних алгоритмів машинного навчання і комп'ютерного зору. Алгоритми OpenCV застосовують у таких сферах:

- Аналіз та обробка зображень
- Системи з розпізнавання обличчя
- Ідентифікації об'єктів
- Розпізнавання жестів у відео
- Відстеження переміщення камери

- Побудова 3D моделей об'єктів
- Створення 3D хмар точок з стерео камер
- Склеювання зображень між собою, для створення зображень всієї сцени з високою роздільною здатністю
- Система взаємодії людини з комп'ютером
- Пошуку схожих зображень із бази даних
- Усування ефекту червоних очей при фотозйомці із спалахом
- Стеження за рухом очей
- Аналіз руху
- Ідентифікація об'єктів
- Сегментація зображень
- Трекінг відео
- Розпізнавання елементів сцени і додавання маркерів для створення доповненої реальності

та інші.

4.2 Структура програмного засобу

Програмний засіб складається з трьох функціональних модулів, двох допоміжних модулів та трьох зберігаючих модулів. Всі ці модулі об’єднує основний, який зветься Application. Модуль Application також зберігає у собі логіку перемикання активного вікна.

Функціональні модулі:

- Capture – модуль, що відповідальний за взяття та збереження фреймів з відеопотоку.
- FaceFeatureDetector – модуль, що відповідальний за знаходження частин обличчя на зображенні та їх класифікацію.

- FaceFeatureTracker – модуль, що відповідальний за знаходження особливих точок в області очей людини.

Допоміжні модулі:

- Configuration – зберігає всю необхідну інформацію для коректування роботи системою (номер користувача, роздільну здатність камери, параметри моніторів, тощо).
- Constants – зберігає основні константи що використовуються в програмі.

Зберігаючи модулі:

- FaceGeometry – зберігає у себе всі необхідні дані про частини обличчя та їх розташування. Також проводить прості геометричні перетворення.
- PoseEstimationHelper – зберігає змінні дані, що необхідні для обчислення положення голови у просторі.
- Pose – зберігає дані про положення голови користувача у просторі.

4.2.1 UML діаграма програми

На рисунку 4.1 наведена UML діаграма класів програми.

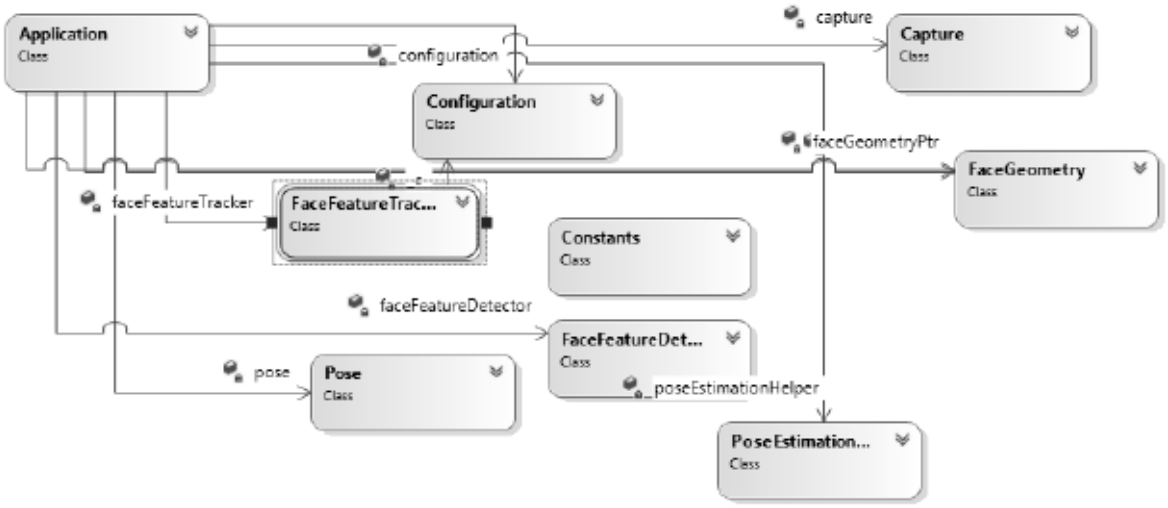

Рисунок 4.1 – UML діаграма класів програмного засобу

4.2.2 Алгоритм роботи програми

Перше, що необхідно знати – це положення моніторів відносно камери. Це задається через спеціальний конфігураційний файл про положення моніторів (monitor_config.ini).

Для захвата зображення з відеопотоку було застосовано функцію Capture::getFrame(), використовуючи бібліотеку OpenCV.

Далі, для знаходження обличчя на зображенні використовується функція cvHaarCascade(), що базується на алгоритмі Віола-Джонса.

Як тільки обличчя було знайдено, йде пошук частин обличчя за вибраним методом пошуку шаблонів. Функція що визивається – FaceFeatureTracker::process().

Для виконання цього методу необхідно мати шаблони цих частин обличчя для роботи алгоритму, що знаходяться за шляхом %EXE_FOLDER%/%PROFILE_NUMBER%/profile/__FACE_PART__.jpg.

Використовуючи перспективну інверсію за трьома точками, знаходиться вектор напряму голови людини та її положення у просторі відносно камери. Ці дані зберігаються у модулях FaceGeometry та Pose.

Необхідно знати фокальну відстань камери та антропоморфні дані голови людини для вирішення проблеми інверсії. Антропоморфні дані можуть використовуватися середньостатистичні (як це вбудовано в програмі за замовченням), або може бути введеним користувачем в якості конфігураційного файлу (face_config.ini). Ці дані зберігаються у модулі Config.

Після цього йде пошук особливих точок ока за допомогою функції FaceFeatureDetector::Process(). Ці дані зберігаються у модулі FaceGeometry.

Для цього також необхідні шаблони тих місць, що нас цікавлять. До того ж кожний з цих шаблонів повинен мати якірну точку, для того щоб знайти особливу точку відносно вихідного зображення. Шлях до файлу, що описує

якірні точки для визначених шаблонів наступний: %EXE_FOLDER%/%PROFILE_NUMBER%/profile/anchors.txt

Як тільки були знайдені особливі точки ока, визначається напрямок погляду людини за кожним оком окремо за допомогою простих геометричних перетворень у модулі FaceGeometry. Тут також використовується середньостатистичні дані за розміром очного яблука.

При відомому положенні голові у просторі, а також положення очей та напрям фокусування погляду, можна визначити точку перетину з площиною, в якої знаходиться монітор.

Для знаходження останнього активного вікна використовується метод Application::getLastActiveWindow(int monitorID). Далі йде перемикання вікна за допомогою функції Application::changeOnWindow(char * windowName) та перенесення курсору за допомогою std::moveCursor(x,y).

Схему, що описує не деталізований алгоритм та взаємодію частин алгоритму між собою, зображена на рисунку 4.2.

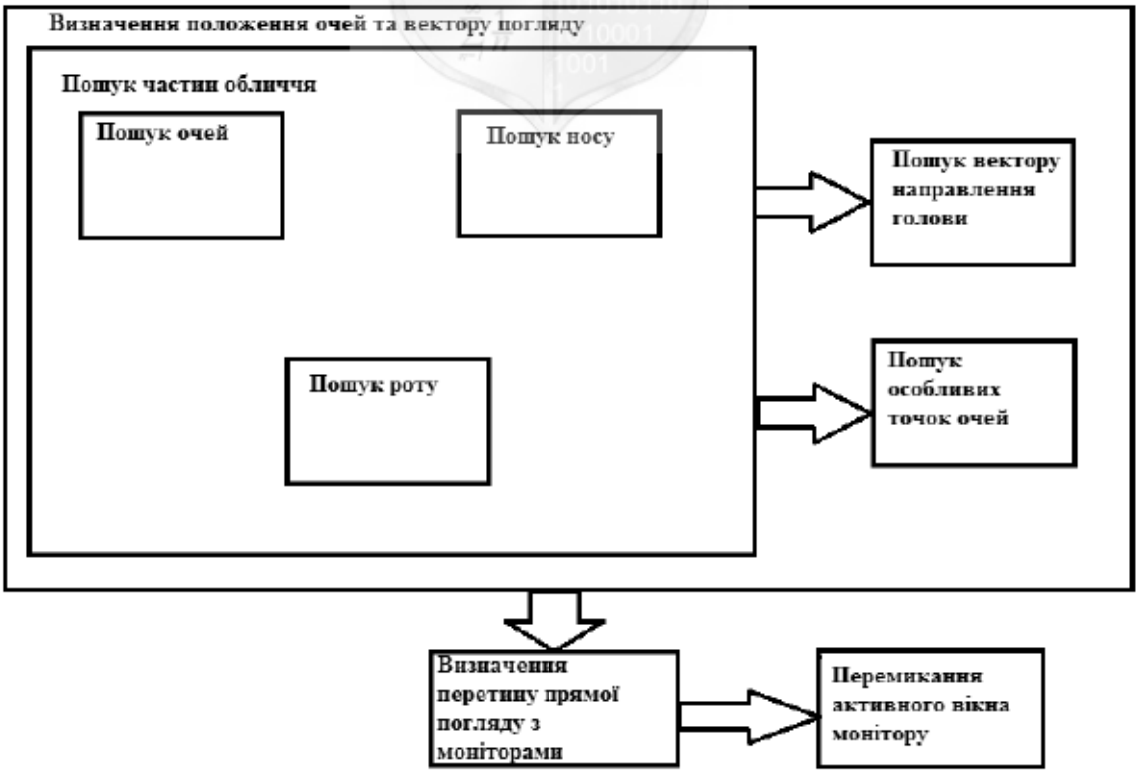

Рисунок 4.2 – Схема взаємодії під процесів процесу відстеження фокусування погляду людини на моніторах та перемикання активного вікна

4.3 Керівництво користувача

Для запуску програми та роботи з нею потрібно встановити наступне програмне забезпечення та мати такі характеристики системи:

- Середовище Visual Studio 2013;
- OpenCV 3.0;
- Система Windows 8.1;
- Вбудована камера з роздільною здатністю не менше 1 МП;

Для початку роботи з програмою варто запустити файл з розширенням .exe. Для того, щоб занести дані, що програма виводить, у файл, необхідно запустити програму у наступному форматі:

Program.exe > PATH_TO_FILE/FILE_NAME.txt

Програма не має специфічних режимів для визначення ролей користувачів, тобто працює лише в одному режимі. Після запуску програми користувачу необхідно зачекати доки програма не підключиться до відеопотоку.

Якщо користувач хоче вийти з програми, він має натиснути клавішу «Esc».

Якщо користувач бажає зробити скріншот програми, він може натиснути клавішу «Enter».

Як тільки користувач запустив програму, перед ним з'являються два вікна: консольне вікно, що що виводить актуальну інформацію (про положення голови, очей, статус активного вікна та монітору, тощо)(рис. 4.3) та вікно з графічним відображенням результату роботи програми(рис. 4.4). Код програми та її модулів можна знайти у додатку Б.

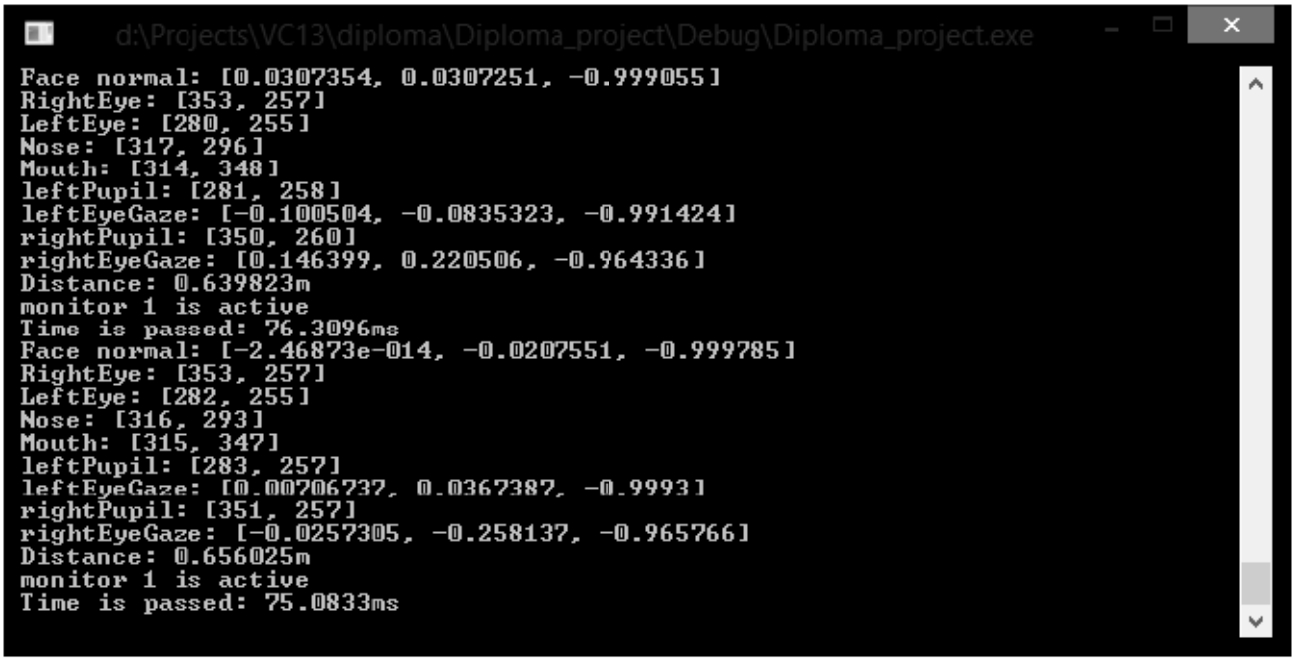

Рисунок 4.3 – Консольне вікно з виведенням актуальної інформації

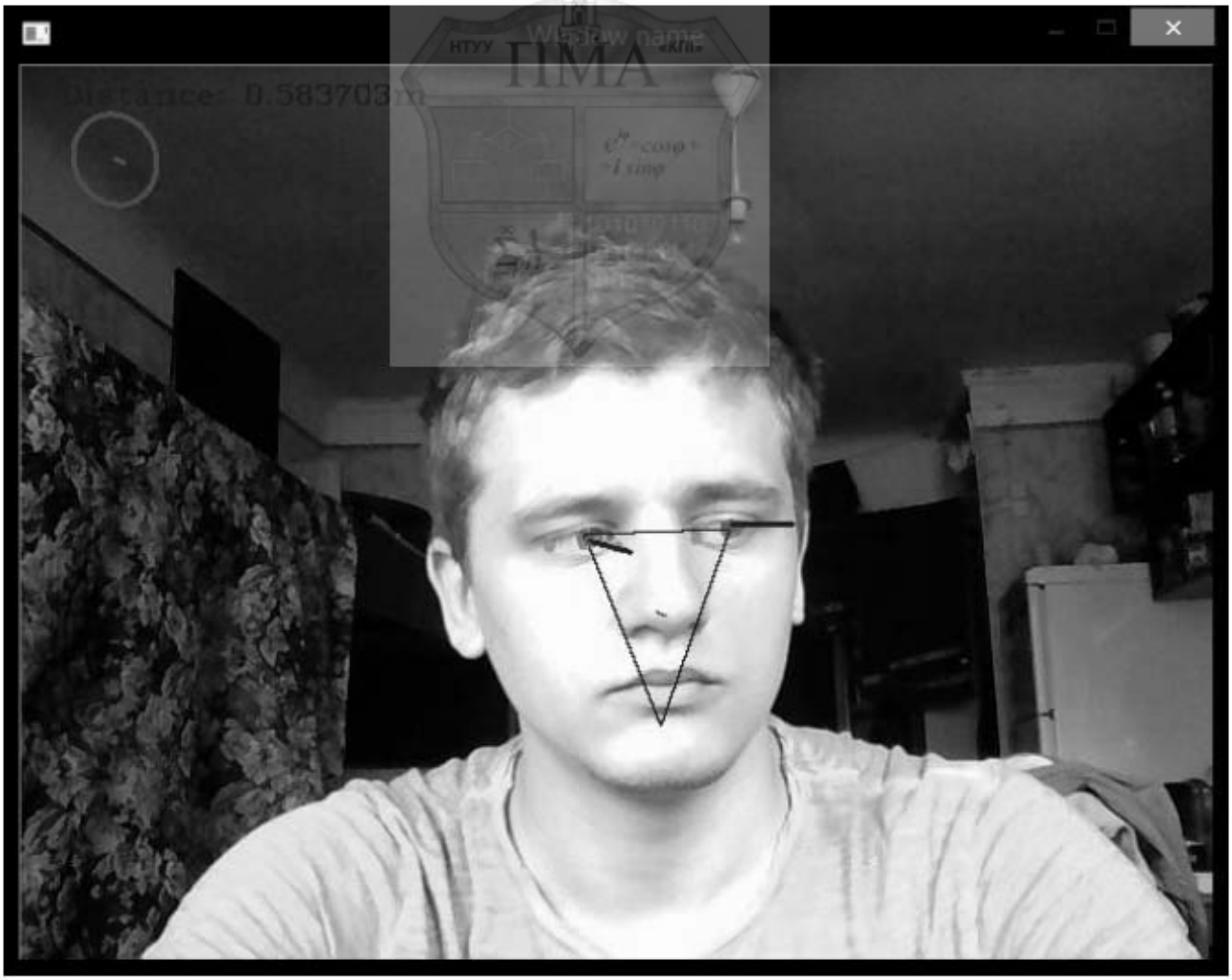

Рисунок 4.4 – Графічне вікно з візуальним відображенням результату роботи програми

4.4 Тестування програмного засобу

Так як час виконання програми та точність визначення особливих точок завжди задовілняли початкові вимоги від програмного засобу при роботі з програмою, то тестування програми в основному проводиться на визначення збігу обрахованої точки перетину вектору фокусування погляду людини з одним з моніторів. Для більш детального аналізу подалі було вирішено розбити тестування на три етапи:

- Тестування визначення положення голови людини.
- Тестування компоненту моніторингу фокусування погляду людини.
- Тестування компоненту перевірки перетину вектору погляду людини з моделлю моніторів.

4.4.1 Тестування визначення положення голови людини

Тестування цієї стадії проводилось наступним чином:

- Респондент сідає на відстані від монітору в 50, 60, 70, 80 см.
- Респондент повертає голову на 30 градусів догори, донизу, наліво та направо (для цього розташовувалися спеціальні маркери, куди необхідно напрямляти голову).
- Респондент повертає голову, напрямляючи її на фіксовані точки (кути монітору).
- Респондент дивиться прямо на камеру.

Результати тестування цієї стадії можна знайти за таблицею 4.1. Значення в таблиці позначають середнє відхилення від зазначеного кута до десятої. Також система намагається обрахувати відстань респондента до камери. Відхилення відстані до камери визначається у відсотках.

Таблиця 4.1 – Результати тестування визначення положення голови людини.

	50 см	60 см	70 см	80 см
Лівий верхній кут	0.3	0.6	1.2	2.0
Правий верхній кут	0.5	0.6	1.0	1.8
Лівий нижній кут	0.4	0.5	1.1	1.5
Лівий верхній кут	0.5	0.7	1.2	2.1
30° вліво	0.8	1.2	1.8	2.5
30° вправо	0.9	1.1	1.7	2.7
30° донизу	0.9	1.2	1.9	2.8
30° доверху	0.8	1.2	1.8	2.7
Прямо	0.1	0.1	0.1	0.1
Знаходження відстані	0.9%	1.3%	1.8%	2.9%

4.4.2 Тестування компоненту моніторингу фокусування погляду людини

Тестування цієї стадії проводилось наступним чином:

- Респондент сідає на відстані від монітору в 50, 60, 70, 80 см.
- Голова респондента зафіксована та напрямлена на камеру, респондент дивиться на фіксовані точки (кути монітору).
- Голова респондента не зафіксована, респондент дивиться на фіксовані точки (кути монітору).

Результати тестування цієї стадії можна знайти за таблицею 4.2 для фіксованого положення голови та за таблицею 4.3 для нефіксованого положення голови. Значення в таблиці позначають середнє відхилення від зазначеного кута до десятої.

Таблиця 4.2 – Результати тестування компоненту моніторингу фокусування погляду людини при фіксованому положенні голови.

	50 см	60 см	70 см	80 см
Лівий верхній кут	5.1	6.3	7.5	8.9
Правий верхній кут	5.3	6.3	7.3	9.2
Лівий нижній кут	4.7	6.4	7.4	9.9
Лівий верхній кут	4.9	6.2	7.5	9.7

Таблиця 4.3 – Результати тестування компоненту моніторингу фокусування погляду людини при вільному положенні голови

	50 см	60 см	70 см	80 см
Лівий верхній кут	15.5	21.2	27.3	32.1
Правий верхній кут	15.7	22.1	26.3	31.8
Лівий нижній кут	15.1	20.4	28.7	34.6
Лівий верхній кут	15.9	19.2	27.5	35.2

4.4.3 Тестування компоненту перевірки перетину вектору погляду людини з моделлю моніторів

Тестування проводилось перед двома моніторами. Респондент повинен дивитися на точки, що з'являються на моніторі в хаотичному порядку. Ці точки рівномірно розподілені на площині монітору. Нижче наводиться рисунок 4.5, що позначає області збігу значення активного монітору у програмі та точками, що з'являлися. Чорний колір – збіг складає більше 90 відсотків, темно-сірий колір – збіг складає більше 70 відсотків, світло-сірий колір – збіг складає більше 40 відсотків.

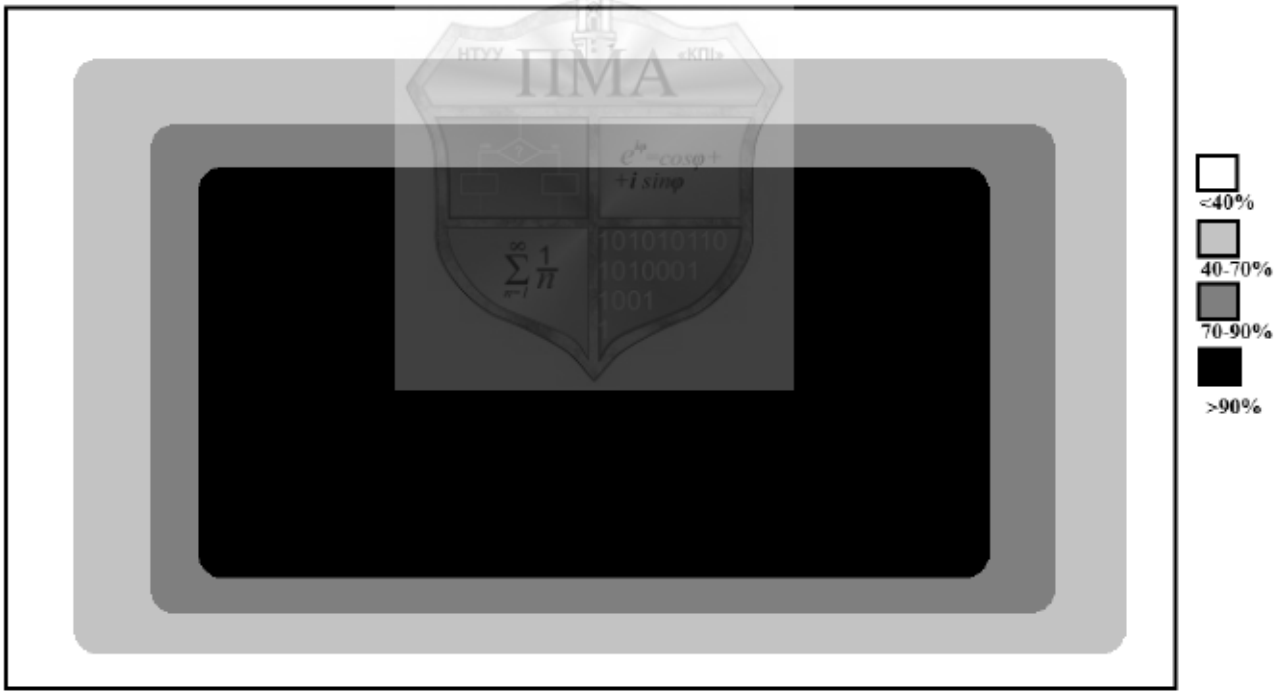

Рисунок 4.5 – Площина монітору з областями збігу та їх частотою

Висновки до розділу 4

В якості мови програмування була обрана мова C++. Це пов'язано з тим, що описана математична модель краще реалізується в об'єктно-орієнтованому підході та дана мова програмування є достатньо швидкою порівняно з іншими мовами на ОС Windows.

Вибір середовища Visual Studio був обумовлений тим, що це середовище є безкоштовне для студентів при використанні у навчальному процесі, а також воно має інструменти для перегляду графічної інформації у режимі налагодження, що значно спростило реалізацію програми.

Було розроблено архітектуру програмного засобу, розглянуті основні його модулі. Для даного програмного засобу розроблено алгоритм, реалізацію якого розглянуто на основі існуючих модулів програми. Схематично змодельовано послідовність всіх процесів у системі. Описано основний функціонал та вимоги для встановлення та використання даного програмного засобу.

Розроблено керівництво користувача з представленням всіх вікон програми. Було проведене тестування системи за трьома стадіями.

При пошуку положення голови людини система дає похибку від 1 до 5 градусів при повороті голові, а також від 1 до 3 відсотків похибку при знаходженні відстані до голови.

При пошуку вектору фокусування погляду людини система дає похибку до 5-10 градусів при фіксованому положенні та до 15-35 градусів при повороті голови. Було припущено, що це пов'язано з низькою роздільною здатністю камери, та, можливо, необхідно подалі шукати способи зменшення цієї похибки.

Так як попередні дві стадії в сумі мають велику похибку (до 35 градусів), це приводить до великої похибки (особливо при повороті голови користувача) при визначення активного монітору, дивлячись на краї монітору.

ВИСНОВКИ

В процесі дослідження були вивчені та проаналізовані основи комп'ютерного зору. Було виявлено, що для вдалого виконання дипломної роботи необхідно знати та використовувати базові методи обробки зображень.

Проаналізувавши сучасні методи обробки зображень, були виділені самі ті, що підходять для обробки відеопотоку в режимі реального часу. Критерій для вибору були наступні: час виконання, складність методів, кількість апаратних ресурсів (пам'ять, процесор), що потребує метод. З цих базових методів складаються більш складніші алгоритми, що допоможуть в вирішенні поставлених підзадач, а саме пошуку обличчя людини, складових обличчя (ніс, око, рот), зіниці.

Проаналізувавши програмні засоби та алгоритми комп'ютерного зору, що виконують схожі поставлені задачі, було прийнято моделювання обличчя людини в 3D форматі на основі знайдених складових обличчя. Це допоможе знайти положення голови користувача у просторі відносно камери, що її фіксує.

В якості основної бібліотеки було обрано OpenCV, що вже має реалізацію оптимізованих базових методів для виконання задачі дипломного роботи.

Програмний засіб робить пошук за допомогою збігання шаблонів з похибкою менше 10 відсотків. Це дозволяє на камері з низькою роздільною здатністю знаходити положення голови у просторі, її напрям та відстань до обличчя.

Швидкість виконання задачі повністю на одному зображенні складає в середньому менше 100 мс, що дозволяє передавати актуальну інформацію в режимі реального часу.

Однак, в ході тестувань програмної моделі були знайдені проблеми з великою похибкою в пошуку напряму зору людини, особливо при позах людини відмінної від фронтової. Насамперед це також пов'язано з низькою роздільною здатністю камери.

Ця робота є початковим етапом до подальшого розвитку програми у направлені комп'ютерного зору та інтерактивних систем з людиною. Подалі планується побудувати систему, що дозволить користуватись комп'ютером виключно за допомогою очей та жестів.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Rogers, Yvonne (2012). "HCI Theory: Classical, Modern, and Contemporary" . Synthesis Lectures on Human-Centered Informatics 5: 1–129. doi:10.2200/S00418ED1V01Y201205HCI014 . Retrieved 7 March 2015.

2. Sengers, Phoebe; Boehner, Kirsten; David, Shay; Joseph, Kaye. "Reflective Design" . CC '05 Proceedings of the 4th decennial conference on Critical computing: between sense and sensibility 5: 49–58. Retrieved 7 March 2015.

3. Blasco JS, Iáñez E, Úbeda A, Azorín J (2012) Visual evoked potential-based brain-machine interface applications to assist disabled people. Expert Systems with Applications 39: 7908–7918. doi: 10.1016/j.eswa.2012.01.110.

4. Gneo, Massimo; Schmid, Maurizio; Conforto, Silvia; D'Alessio, Tommaso (2012). "A free geometry model-independent neural eye-gaze tracking system". Journal of NeuroEngineering and Rehabilitation 9 (1): 82. doi:10.1186/1743-0003-9-82.

5. Sigut, J; Sidha, SA (February 2011). "Iris center corneal reflection method for gaze tracking using visible light.". IEEE transactions on bio-medical engineering 58 (2): 411–9.

6. Hua, H; Krishnaswamy, P; Rolland, JP (15 May 2006). "Video-based eyetracking methods and algorithms in head-mounted displays.". Optics express 14 (10): 4328–50.

7. S. Weston and K. Oleg, "Real-time eye gaze tracking with an unmodified commodity webcam employing a neural network", Conference on Human Factors in Computing Systems-Proceedings, no. 28, (2010), pp. 3739-3744.

8. Л. Шапиро, Дж. Стокман. Компьютерное зрение = Computer Vision. — М.: Бином. Лаборатория знаний, 2006. — 752 с. — ISBN 5-94774-384-1.

9. Дэвид Форсайт, Жан Понс. Компьютерное зрение. Современный подход = Computer Vision: A Modern Approach. — М.: «Вильямс», 2004. — 928 с. — ISBN 5-8459-0542-7.

10. А.А. Лукьяница ,А.Г. Шишкин. Цифровая обработка видеоизображений. — М.: «Ай-Эс-Эс Пресс», 2009. — 518 с. — ISBN 978-5-9901899-1-1.

11. Michael Sapienza and Kenneth P. Camilleri, “Fasthpe: A recipe for quick head pose estimation”, Department of Systems & Control, University of Malta, 2011.

12. M.A. Fischler and R.C. Bolles. The random sample consensus set: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, 24(6):381--395, 1981.

13. Jamil Draréni, “A simple oriented mean-shift algorithm for tracking”, 2007

14. P. Viola and M.J. Jones, «Robust real-time face detection», International Journal of Computer Vision, vol. 57, no. 2, 2004.

15. P. Viola and M.J. Jones, «Rapid Object Detection using a Boosted Cascade of Simple Features», proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2001), 2001.

16. R. Brunelli, Template Matching Techniques in Computer Vision: Theory and Practice, Wiley, ISBN 978-0-470-51706-2, 2009.

17. T.F. Cootes and C.J. Taylor and D.H. Cooper and J. Graham (1995). "Active shape models - their training and application".

18. T.F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. ECCV, 2:484–498, 1998.

19. Zhao, Liang, G. Pingali and I. Carlbom, “Real-time Head Orientation Estimation Using Neural Networks,” Proceedings of International Conference on Image Processing, Vol.1, pp.297-300, 2002.

20. Richard O. Duda and Peter E. Hart "Use of the Hough Transformation to Detect Lines and Curves in Pictures", 1971.

21. Stroustrup, Bjarne (1997). "1". The C++ Programming Language (Third ed.). ISBN 0-201-88954-4. OCLC 59193992.

22. Офіціальний сайт Visual Studio = Visual Studio Official Site[Електроний ресурс]. – Режим доступу: http://www.visualstudio.com .

23. Сайт розробників бібліотеки OpenCV = OpenCV Developer Site[Електроний ресурс] . – Режим доступу: http://code.opencv.org .