Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально-науковий інститут комп'ютерних наук
та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

Магістерська кваліфікаційна робота

другий (магістерський)
(рівень вищої освіти)

на тему:

Віртуальна клавіатура, керована оком, для людей з обмеженими фізичними можливостями

Виконав: студент VI курсу групи КН-62М
спеціальності
122 "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)

Щербань Олег Ігорович
(прізвище та ініціали)

Керівник Яцишин С.І.
(прізвище та ініціали)

Рецензент Карашецький
(прізвище та ініціали)

Львів – 2025

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ комп'ютерних наук та інформаційних технологій
Кафедра комп'ютерних наук
Рівень вищої освіти другий (магістерський)
Спеціальність 122 "Комп'ютерні науки"
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук
_____ Борецька І.Б.
"10" грудня 2025 року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Щербань Олег Ігорович
(прізвище, ім'я, по батькові)

1. Тема роботи: Віртуальна клавіатура, керована оком, для людей з обмеженими фізичними можливостями
Керівник роботи *Яцишин С.І., к.т.н., доцент*,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "29" квітня 2025 року №С-288
2. Термін подання студентом роботи. 10.12.2025
3. Вихідні дані до роботи: створення віртуальної клавіатури, керованої оком, для людей з обмеженими фізичними можливостями
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Розділ 1. Стан проблемної області
Розділ 2. Інформаційне забезпечення
Розділ 3. Математичне забезпечення
Розділ 4. Програмне забезпечення
Розділ 5. Розроблення стартап-проєкту
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Підготовка матеріалу до доповіді
6. Дата видачі завдання. 01.05.2025

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1.	Проаналізувати існуючі рішення для керування комп'ютером за допомогою очей.	06.05.2025	Виконано
2.	Розробити просту модель, яка перетворює рухи очей у вибір клавіш на екрані.	20.05.2025	Виконано
3.	Створити прототип віртуальної клавіатури з графічним інтерфейсом.	14.06.2025	Виконано
4.	Налаштувати інтеграцію з веб камерою та забезпечити калібрування для різних користувачів.	24.06.2025	Виконано
5.	Провести тестування прототипу та зібрати дані про точність і швидкість роботи.	30.06.2025	Виконано
6.	Сформулювати висновки та рекомендації для подальшого удосконалення та впровадження.	10.12.2025	Виконано

Студент _____ Шербань О.І.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Яцишин С. І.
(підпис) (прізвище та ініціали)

Магістерська робота містить 76 сторінок пояснювальної записки, 15 рисунків, 9 таблиць, 1 додаток, 36 використаних джерел.

У роботі розроблено інформаційно-алгоритмічну модель та програмну концепцію віртуальної клавіатури, керованої рухом очей, призначеної для людей з обмеженими фізичними можливостями. Система забезпечує отримання та фільтрацію координат погляду, визначення фіксацій, вибір символів та формування тексту в реальному часі. Запропоновано методи обробки руху очей, включно зі згладжуванням, фільтрацією шумів і комбінованим визначенням фіксацій, а також архітектуру програмних модулів, що узгоджують усі етапи введення тексту. Система може бути використана як у домашніх умовах, так і в реабілітаційних центрах для забезпечення доступної взаємодії з комп'ютером для користувачів з тяжкими порушеннями моторики.

Ключові слова: віртуальна клавіатура, погляд, керування очима, відстеження руху очей, фіксації, інклюзія, допоміжні технології.

ABSTRACT

The thesis contains 76 pages of explanatory note, 15 figures, 9 tables, 1 appendice, and 36 referenced literary sources.

This work presents the development of an information–algorithmic model and a software concept for an eye-controlled virtual keyboard designed for users with limited motor abilities. The system processes raw gaze coordinates, filters noise, detects fixations, and performs symbol selection to enable real-time text input. Advanced gaze-processing methods, including smoothing, noise reduction, and combined fixation detection, are proposed. The software architecture integrates all stages of gaze-based interaction, ensuring accurate and stable operation. The developed system can be used both at home and in rehabilitation centers to provide accessible computer interaction for people with severe motor impairments.

Keywords**:** virtual keyboard, eye tracking, gaze control, fixation detection, accessibility, assistive technologies.

ЗМІСТ

ВСТУП

Актуальність теми. Сучасні тенденції цифрової інклюзії вимагають розроблення інтерфейсів, здатних забезпечити доступ до інформаційних технологій для користувачів з обмеженими моторними можливостями. Традиційні пристрої введення, такі як миша й клавіатура, у багатьох випадках виявляються недоступними, що створює суттєві бар'єри у комунікації, навчанні та професійній діяльності. Це формує стійкий попит на альтернативні, безконтактні засоби взаємодії з комп'ютерними системами.

Технології відстеження погляду, які останніми роками значно вдосконалилися завдяки розвитку комп'ютерного зору, створюють передумови для реалізації ефективних інтерфейсів введення, що не потребують фізичних зусиль. Погляд є одним із найбільш стабільних моторних каналів, який часто зберігається навіть за умов тяжких порушень рухливості. Тому віртуальна клавіатура, керована рухом очей, виступає перспективним інструментом забезпечення доступності та автономності для широкої групи користувачів.

Попри наявні комерційні рішення, існує потреба у створенні точних, адаптивних і програмно незалежних систем, здатних працювати у реальному часі та враховувати індивідуальні особливості рухів очей. У цьому контексті науково-практичне значення має розроблення моделі обробки координат погляду та алгоритмів вибору символів, придатних для застосування у віртуальних клавіатурах, оптимізованих для людей з обмеженою рухливістю.

Мета і завдання дослідження

Мета роботи: розробити інформаційно-алгоритмічну модель і програмну концепцію віртуальної клавіатури, керованої поглядом, що забезпечує точне та стабільне введення тексту користувачами з порушеннями моторики.

Для досягнення мети поставлено такі завдання:

1. Проаналізувати сучасні системи відстеження руху очей та існуючі інтерфейси введення, орієнтовані на користувачів з обмеженою рухливістю.

2. Побудувати інформаційну модель процесу введення тексту за допомогою погляду.
3. Розробити та обґрунтувати методи фільтрації координат погляду, виявлення фіксацій та алгоритми вибору символів.
4. Сформувати програмну архітектуру системи й визначити вимоги та обмеження її роботи.
5. Оцінити придатність запропонованих рішень для використання у системах допоміжних технологій.

Об’єкт дослідження: процес взаємодії користувача з комп’ютерною системою через аналіз руху очей.

Предмет дослідження: методи обробки координат погляду та алгоритми вибору символів у віртуальній клавіатурі.

Практичне значення роботи полягає у можливості використання розробленої моделі та алгоритмів для створення реальних систем введення тексту поглядом. Це рішення може застосовуватися вдома та в реабілітаційних центрах, забезпечуючи доступну комунікацію для людей з порушеннями моторики.

Наукова новизна роботи полягає у:

- формуванні комбінованої моделі визначення фіксацій, що поєднує швидкісні та дисперсійні критерії й адаптується до індивідуальних особливостей користувача;
- розробленні інформаційної структури системи введення тексту, яка забезпечує узгоджену обробку координат, фіксацій та подій вибору символів у реальному часі;
- запропонуванні архітектурного підходу до побудови віртуальної клавіатури, оптимізованої для керування поглядом у користувачів з порушеннями рухливості.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1. Актуальність технологій керування поглядом

Технології керування поглядом стрімко набирають поширення у сферах, де традиційні методи взаємодії з комп'ютером є недоступними або обмеженими. Зростання кількості людей, які не можуть повноцінно використовувати мишу чи клавіатуру через захворювання, травми або вроджені порушення, формує потребу у засобах, що дозволяють виконувати звичні цифрові дії без фізичного навантаження. У цьому контексті напрям, пов'язаний із відстеженням руху очей, перетворюється на реальну можливість забезпечити доступ до комп'ютерів для користувачів з різними формами моторних порушень.

Погляд є одним із найбільш стабільних каналів комунікації, який зберігається навіть за умови тяжких порушень рухливості. На відміну від жестових або голосових систем, здатність контролювати рух очей, як правило, зберігається у більшості пацієнтів із нейром'язовими або дегенеративними захворюваннями. Це робить технології інтерфейсу доцільним інструментом у побудові альтернативних методів керування комп'ютерними системами [1].

Віртуальні клавіатури, що реагують на напрям погляду, створюють умови для виконання базових та розширених дій від введення тексту до запуску програм. Вони компенсують неможливість використання фізичних пристроїв введення, зберігаючи доступ до цифрового середовища. У таких системах погляд виступає не лише індикатором присутності користувача, а й точним інструментом керування, здатним замінити дотик. Сучасні технології відстеження погляду досягли рівня, що дозволяє аналізувати фіксації очей із високою точністю навіть у змінних умовах освітлення. Це створює технічну основу для побудови стабільних інтерфейсів введення тексту та взаємодії з меню. Завдяки цьому віртуальні клавіатури стають доступнішими та придатними для щоденного використання вдома, у реабілітаційних центрах і лікарнях.

Для людей з обмеженими фізичними можливостями можливість самостійно вводити текст відіграє значну роль у збереженні комунікаційних навичок. Відсутність

здатності писати або друкувати суттєво ускладнює спілкування та соціальну взаємодію, що часто призводить до ізоляції. Тому технології керування поглядом розглядаються як засіб підтримки незалежності користувача, дозволяючи йому самостійно листуватися, працювати з документами та отримувати інформацію. Особливо актуальними такі рішення є для людей з наслідками інсульту, ДЦП, травмами спинного мозку або розсіяним склерозом. У цих випадках традиційні інтерфейси втрачають свою ефективність через обмеження рухів рук або загальну слабкість м'язів. Системи керування поглядом забезпечують можливість відновлення частини втрачених функцій, відкриваючи шлях до самостійного виконання рутинних дій.

Технології Eye Tracking активно впроваджуються у сфери, де важливе мінімальне фізичне навантаження. На відміну від голосових інтерфейсів, вони не залежать від чіткості вимови або сторонніх шумів, що робить їх стабільнішими в умовах лікарняних палат, реабілітаційних кабінетів або домашнього середовища. Це формує широкі можливості для побудови універсальних рішень, орієнтованих на різні групи користувачів. З технічного погляду сучасні системи відстеження погляду інтегрують алгоритми комп'ютерного зору, які здатні працювати у режимі реального часу. Це забезпечує плавність та природність взаємодії з віртуальною клавіатурою, що є важливим для досягнення ефективної швидкості введення тексту. Покращення стабільності таких алгоритмів зменшує кількість хибних спрацьовувань, що підвищує точність введення [2].

Існує тенденція до поєднання відстеження погляду з інтелектуальними методами прогнозування тексту. Це дозволяє компенсувати повільнішу швидкість друку, властиву ocul-based системам. Завдяки побудові ймовірнісних моделей користувач може вводити фрази та слова із меншими витратами часу, що робить такі системи придатними не лише для базових, а й для тривалих сеансів роботи.

У реабілітаційних практиках технології керування поглядом виконують подвійну функцію: вони забезпечують засіб комунікації та водночас можуть використовуватися як інструмент тренування зорово-моторної координації. Застосування віртуальних клавіатур дозволяє фахівцям адаптувати середовище під

стан пацієнта та поступово збільшувати складність завдань, що сприяє відновленню навичок контролю уваги та фокусування. В умовах розвитку цифрової інклюзії поширюються програмні рішення, які надають користувачам розширені можливості незалежно від їх фізичних обмежень. Віртуальна клавіатура, керована поглядом, відповідає цьому напрямку, оскільки гарантує доступ до комп'ютера без потреби у спеціальних фізичних пристроях. Це підвищує доступність цифрових сервісів та дозволяє людям брати участь у навчанні, роботі та соціальних активностях.

У науковій спільноті спостерігається інтерес до вдосконалення способів інтерпретації руху очей. Це пов'язано з тим, що поведінка погляду містить інформацію не лише про напрям уваги, а й про намір користувача. Тому системи керування поглядом усе частіше розглядаються як повноцінні канали введення команд, а не як допоміжні засоби [3].

Підвищення точності камер та пришвидшення обробки зображення дозволили створювати недорогі рішення, які можуть працювати на звичайних комп'ютерах або ноутбуках. Завдяки цьому відпадає потреба у дорогих спеціалізованих трекерах, що робить технології доступнішими для домашнього використання. Це суттєво впливає на можливість широкого впровадження таких систем у повсякденне життя користувачів.

Технології керування поглядом активно інтегруються з іншими допоміжними технологіями екранними читачами, системами автоматичного наведення, прогнозуванням тексту та мікроруховими фільтрами. Поєднання цих елементів дає змогу будувати комплексні інтерфейси, зручні для користувачів із різними потребами. У межах таких систем віртуальна клавіатура стає незамінним інструментом введення інформації. Сучасні державні та міжнародні програми, спрямовані на підвищення цифрової доступності, також стимулюють розвиток рішень на основі керування поглядом. Упровадження таких систем у лікарнях, соціальних установах та навчальних закладах сприяє інтеграції людей з обмеженими фізичними можливостями у цифровий простір. Це формує потребу у більш точних, зручних та адаптивних інтерфейсах, серед яких віртуальні клавіатури на основі руху очей займають помітне місце.

1.2. Аналіз сучасних систем відстеження руху очей

Сучасні системи відстеження руху очей ґрунтуються на комбінації апаратних та програмних компонентів, що дозволяють визначати положення очей та напрям зору у реальному часі. Основою більшості рішень є методи комп'ютерного зору, що аналізують зображення зі звичайних або інфрачервоних камер. Висока частота зчитування та точність обробки дозволяють формувати стабільний сигнал для керування інтерфейсами, у тому числі віртуальними клавіатурами. На ринку існують два основні підходи до створення трекерів: апаратні пристрої, виконані у вигляді спеціалізованих модулів, та програмні системи, що використовують камери загального призначення. Апаратні трекери зазвичай оснащені інфрачервоним підсвічуванням і кількома сенсорами, що дає змогу зменшити кількість шумів та забезпечити стабільність за різних умов освітлення. Програмні рішення покладаються на алгоритми компенсації освітлення та цифрові методи покращення якості зображення, що робить їх доступнішими та дешевшими [4].

Одним із найпоширеніших форматів є системи, які використовують інфрачервоні світлодіоди для освітлення очей та виділення відблисків на рогівці. Ці відблики дозволяють алгоритмам точно розпізнавати координати погляду відносно екрана. Такий підхід забезпечує високу точність, однак потребує спеціального обладнання. Для людей з обмеженою рухливістю це часто є перевагою, оскільки трекер працює стабільно у фіксованому положенні. Програмні системи на основі звичайних вебкамер стали окремим напрямом, що отримав широке застосування завдяки доступності споживчих пристроїв. Вони використовують методи детекції обличчя, локалізації зіниці, аналізу форми повік та оцінки руху очей. Незважаючи на нижчу точність порівняно з інфрачервоними трекерами, такі системи демонструють достатній рівень якості для побудови інтерфейсів базового керування, зокрема для введення тексту у віртуальних клавіатурах.

Відомі комерційні системи, такі як Tobii, EyeTech чи Pupil Labs, орієнтовані на різні категорії користувачів. Tobii забезпечує дуже високу частоту зчитування та

надійний аналіз погляду, що робить його придатним для наукових досліджень і систем підтримки людей з інвалідністю. Pupil Labs пропонує модульні рішення з відкритою архітектурою, що дає змогу адаптувати алгоритми під конкретні завдання. Ці системи демонструють, як різні підходи до проектування впливають на баланс між точністю, швидкодією та вартістю. Окрему групу становлять трекери, інтегровані у VR- та AR-пристрої. У таких системах аналіз погляду виконується не лише для керування, а й для оптимізації рендерингу та побудови індивідуального профілю роботи з віртуальним середовищем. Використання кількох камер, розташованих всередині гарнітури, забезпечує точне виявлення рухів очей. Проте ці рішення не завжди підходять для користувачів із серйозними фізичними порушеннями, оскільки потребують носіння важкої гарнітури.

Сучасні системи відстеження погляду часто поєднують декілька аналітичних алгоритмів: визначення центру зіниці, оцінку векторів погляду, аналіз фіксацій і саккад. Комбінація цих методів дозволяє більш точно відтворювати наміри користувача та зменшувати кількість хибних спрацьовувань. Це особливо актуально для систем, що застосовуються у засобах комунікації для людей з обмеженою рухливістю, де точність введення безпосередньо впливає на комфорт використання.

Значна частина сучасних досліджень спрямована на покращення точності за допомогою моделей глибокого навчання. Нейронні мережі здатні враховувати індивідуальні особливості обличчя та очей, що робить системи більш стійкими до змін освітлення, положення голови та шумів. Використання таких моделей дає змогу будувати віртуальні клавіатури, які реагують швидше та здатні адаптуватися до конкретного користувача. У практиці адаптивних інтерфейсів поширені методи фільтрації мікрорухів та згладжування траєкторій погляду. Це допомагає компенсувати природну нестабільність рухів очей та дозволяє виконувати точні вибори символів на клавіатурі. Такі техніки особливо актуальні для людей, які не можуть довго утримувати фіксацію через неврологічні порушення або втому [4]. Системи відстеження погляду у медичних умовах часто доповнюються функціями калібрування, що дозволяють налаштувати чутливість під конкретного користувача. Правильне калібрування знижує навантаження на користувача під час роботи з

віртуальною клавіатурою та сприяє більш точному введенню інформації. Для людей із тяжкими порушеннями рухливості процедура калібрування є одним з ключових етапів адаптації системи.

У контексті створення допоміжних технологій велике значення має здатність системи працювати у реальному часі без затримок. Навіть незначні затримки у відображенні погляду можуть суттєво знизити ефективність введення тексту. Тому сучасні системи орієнтовані на досягнення мінімальної латентності обробки сигналів, що забезпечує плавну взаємодію з віртуальною клавіатурою.

Проаналізувавши сучасні системи відстеження погляду, можна зробити висновок, що розвиток цього напряму рухається у бік доступності, точності та адаптивності. Апаратні рішення забезпечують максимальну стабільність, тоді як програмні системи відкривають шлях до повсюдного використання завдяки низькій вартості. Поєднання апаратних і програмних методів створює умови для розроблення інтерфейсів, здатних задовольнити потреби людей з різними рівнями рухових можливостей, що прямо підтверджує доцільність використання таких технологій у рамках віртуальної клавіатури, керованої поглядом.

1.3. Огляд існуючих віртуальних клавіатур і технологій доступності

Віртуальні клавіатури стали основою багатьох систем альтернативного введення тексту, насамперед для користувачів, які не можуть використовувати фізичні кнопкові пристрої. Перші програмні клавіатури були орієнтовані на введення символів за допомогою миші або стилуса, проте з розвитком технологій вони адаптувалися до ширшого кола засобів керування, включно з трекерами погляду, сенсорними екранами, голосовими командами та пристроями з фіксатором єдиного натискання. Це зробило їх універсальним інструментом у середовищах, де важливо забезпечити доступність комп’ютерних систем.

Одним із найвідоміших рішень є екранна клавіатура Windows On-Screen Keyboard, що входить до стандартних засобів операційної системи Windows. Вона забезпечує базову функціональність і може активуватися через будь-який пристрій

введення. Хоча її можливості є обмеженими, вона слугує прикладом мінімально необхідного функціоналу, що дозволяє вводити текст без фізичної клавіатури [5]. Для користувачів з інвалідністю вона часто використовується у поєднанні з окремими програмами відстеження погляду. Більш розвиненим інструментом у середовищі Windows є система Windows Eye Control, яка поєднує екранну клавіатуру, трекінг погляду та можливості навігації інтерфейсом. Це рішення дозволяє вводити текст, переміщувати курсор і активувати елементи через фіксацію погляду. Система адаптована для роботи з декількома відомими трекерами, що робить її зручною у розгортанні як у домашніх умовах, так і у медичних установах. Серед програмних інструментів часто використовується клавіатура Click2Speak, що створена спеціально для людей із порушеннями рухливості. Вона підтримує введення тексту через погляд, перемикачі та інші доступні методи взаємодії. Система включає функції прогнозування тексту, налаштування зовнішнього вигляду та адаптивні затримки фіксації, що зменшує кількість помилкових виборів. Такий підхід демонструє важливість розширеної персоналізації для підвищення зручності роботи.

Помітне місце серед рішень займає платформа Dasher, що використовує інтерфейс на основі жестів і передбачення тексту. Хоча вона суттєво відрізняється від традиційних клавіатур, Dasher надає користувачеві можливість друкувати значно швидше завдяки моделі прогнозування. Вона підтримує контроль поглядом, що дозволяє застосовувати її для людей із тяжкими порушеннями рухливості. Інтерактивність і плавність керування роблять цю платформу ефективною альтернативою класичним клавіатурним розкладкам.

У мобільних системах віртуальні клавіатури набули широкого поширення завдяки сенсорним екранам. Інструменти, такі як Gboard або SwiftKey, включають можливості автодоповнення, автоматичного виправлення та введення жестами. Хоча вони не орієнтовані спеціально на користувачів із фізичними порушеннями, методи адаптивного введення можуть бути корисними у поєднанні з програмами відстеження погляду, встановленими на смартфонах або планшетах [6].

Окрему групу складають системи, розроблені спеціально для людей із тяжкими порушеннями моторики. Наприклад, EyeGaze Edge та інші схожі рішення включають

екранні клавіатури, оптимізовані для повільних і точних фіксацій. Ці платформи передбачають можливість регулювання чутливості, розміру клавіш, яскравості інтерфейсу та інтервалу вибору. Вони забезпечують точність, необхідну для щоденної роботи, включно з веденням документації або спілкуванням через месенджери.

У системах доступності для macOS важливим інструментом є Switch Control, що пропонує використання екранної клавіатури з контролем через один або кілька перемикачів. Хоча ця система не призначена для керування очима, вона ілюструє принцип побудови інтерфейсу, у якому всі функції комп'ютера можуть бути виконані без традиційних пристроїв введення. Багато концепцій Switch Control пізніше адаптувалися для платформ з підтримкою погляду. На ринку також присутні онлайн-клавіатури, що функціонують у браузері без встановлення додаткового ПЗ. Вони корисні для одноразового або тимчасового введення тексту, проте зазвичай не забезпечують достатньої точності та адаптивності для користувачів із інвалідністю. Попри це, вони демонструють напрям розвитку веб-інструментів та потенціал облаштування кросплатформних систем введення.

Помітний розвиток отримали технології прогнозування тексту та автодоповнення, що дозволяють суттєво прискорювати процес введення у системах керування поглядом. Комбінація віртуальної клавіатури та алгоритмів передбачення дає змогу вводити довгі фрази за мінімальної кількості фіксацій. Це підвищує ефективність і зменшує навантаження на користувача, що є особливо важливим у реабілітаційному контексті. У сучасних системах доступності зростає інтерес до інтеграції голосових інтерфейсів, автоматичного читання тексту, навігаційних підказок та адаптивних тем інтерфейсу. Усі ці рішення формують гнучке середовище, яке може підлаштовуватися під різні стани користувача. Синтез методів керування поглядом із додатковими інструментами дає змогу створювати багатофункціональні платформи для взаємодії [7].

Підсумовуючи, можна зазначити, що сучасні віртуальні клавіатури та засоби доступності демонструють широкий діапазон підходів: від простих екранних розкладок до адаптивних систем із розвиненим прогнозуванням і підтримкою трекінгу погляду. Розвиток цих технологій створює умови для появи нових рішень,

які можуть бути оптимізовані спеціально для людей з обмеженими фізичними можливостями. Саме на цьому ґрунтується необхідність подальшого вдосконалення віртуальних клавіатур, здатних забезпечити природне та надійне введення тексту за допомогою руху очей.

Висновки до розділу

Аналіз сучасного стану технологій керування поглядом показує, що цей напрям набув значного розвитку завдяки поєднанню комп'ютерного зору, апаратних сенсорів та інтелектуальних алгоритмів обробки сигналів. Технології, орієнтовані на користувачів із порушеннями рухливості, забезпечують можливість виконувати дії, які раніше були недоступними без фізичної взаємодії з пристроями. Вони стали основою для розроблення інтерфейсів, здатних реагувати на мікрорухи очей, підтримувати введення тексту, забезпечувати навігацію та адаптуватися до індивідуальних характеристик користувача. Розвиток систем, що працюють у реальному часі, а також удосконалення моделей прогнозування значно підвищили ефективність використання таких технологій у побутових та реабілітаційних умовах.

Огляд існуючих віртуальних клавіатур та платформ доступності засвідчив, що на ринку присутні різнорівневі рішення: від базових екранних розкладок до комплексних інтерфейсів з високим ступенем персоналізації. Проте більшість доступних систем або не враховують специфіку введення за допомогою погляду, або не забезпечують достатнього балансу між точністю, швидкодією та зручністю. Це підкреслює потребу у створенні спеціалізованих віртуальних клавіатур, оптимізованих саме для керування очима, здатних забезпечити стабільність роботи та комфорт користувача. Таким чином, результати розділу підтверджують доцільність подальшого дослідження та розроблення власного рішення, орієнтованого на людей з обмеженими фізичними можливостями.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Постановка задачі та опис об’єкта дослідження

Об’єктом дослідження у межах цієї роботи є процес взаємодії користувача з комп’ютерною системою за допомогою руху очей. Такий процес охоплює сприйняття зорової активності сенсорними засобами, аналіз координат погляду, формування керувальних сигналів та їхнє перетворення у дії, що відтворюють механізм введення тексту. Цей об’єкт відображає спосіб, у який користувачі, обмежені у фізичній рухливості, можуть здійснювати повноцінну взаємодію з цифровим середовищем без використання традиційних пристроїв введення.

Дослідження базується на припущенні, що рух очей можна розглядати як стійкий та достатньо точний сигнал для керування віртуальними інтерфейсами. Пристрої відстеження погляду забезпечують отримання координат фіксацій, які можуть використовуватися для вибору елементів клавіатури, переміщення курсора та активування команд. Це створює умови для побудови системи, у якій погляд виконує роль основного каналу введення. У центрі уваги знаходиться віртуальна клавіатура, що реагує на погляд користувача і дозволяє вводити текст з мінімальними фізичними зусиллями. Вона повинна враховувати характерні особливості рухів очей: мікрорухи, саккади, фіксації та природні варіації у положенні голови. Ці особливості визначають вимоги до точності обробки та методів згладжування даних, що надходять від трекера погляду [8].

Проблема, що розглядається у роботі, полягає у необхідності створення інтерфейсу, здатного забезпечити стабільне та зручне введення тексту для людей з тяжкими порушеннями моторики. Традиційні інтерфейси не можуть бути використані у таких умовах, тому виникає потреба у розробленні спеціалізованої системи, що враховує індивідуальні можливості користувача та особливості аналізу рухів очей. Це потребує створення програмного рішення, у якому всі елементи інтерфейсу будуть адаптовані до способу керування поглядом.

Постановка задачі передбачає необхідність формування механізму точного визначення моменту вибору символу. На відміну від традиційного кліка, у системах

керування очима вибір здійснюється за рахунок фіксації погляду на певній кнопці протягом визначеного часу або через використання додаткових жестів очей. Це потребує встановлення параметрів чутливості, що забезпечують баланс між швидкістю введення та кількістю хибних спрацьовувань. Ще одним елементом постановки задачі є формування зручної розкладки клавіатури. Стандартні клавіатури не підходять для керування поглядом через щільне розташування символів. Для користувачів із обмеженими руховими можливостями необхідно збільшувати розмір клавіш, регулювати відстань між ними та адаптувати позиціонування відповідно до характерних траєкторій руху очей. Таким чином, завданням дослідження є створення такої розкладки, що враховуватиме умови введення через погляд.

Опис об’єкта також передбачає врахування умов експлуатації системи. Використання трекера погляду потребує стабільного освітлення, правильного позиціонування користувача та мінімізації факторів, які можуть викликати збої у визначенні фіксацій. Тому система введення має бути здатною працювати у змінних умовах, а алгоритми компенсувати дрібні порушення під час роботи користувача.

Необхідною частиною є реалізація процесу калібрування, що дозволяє системі враховувати індивідуальні параметри користувача. Кожна людина має унікальні особливості будови очей, амплітуди рухів і стилю концентрації уваги, тому калібрування є необхідною умовою для забезпечення точності. У межах цього дослідження калібрування розглядається як процес узгодження координат, отриманих від камери, з координатами елементів віртуальної клавіатури.

Поставлена задача також охоплює аналіз способів збільшення ефективності введення тексту. Для систем, керованих поглядом, темп введення значно нижчий, ніж у традиційних інтерфейсах, тому застосовуються методи прогнозування тексту. Завдання полягає у визначенні способу інтеграції таких методів у віртуальну клавіатуру таким чином, щоб вони зменшували кількість виборів і не ускладнювали інтерфейс. Важливо враховувати, що користувачі з порушеннями рухливості можуть швидко втомлюватися під час тривалих сеансів роботи. Це визначає необхідність оптимізації інтерфейсу таким чином, щоб скоротити кількість рухів очей між клавішами, зменшити тривалість фіксацій та забезпечити можливість регулювання

темпу роботи. Система має підтримувати адаптацію під індивідуальний стан користувача.

Опис об’єкта дослідження також включає характеристики алгоритмів, що визначають фіксації та саккади. Від їхньої точності та стабільності залежить якість введення, а також здатність системи уникати хибних натискань під час руху погляду. Завдання дослідження полягає у виборі та порівнянні алгоритмів, придатних для використання у реальних умовах, де фактори шуму неминучі. Для коректної роботи віртуальної клавіатури потрібен графічний інтерфейс, у якому всі елементи мають чіткі границі, видимість та можливість масштабування. Тому частиною постановки задачі є визначення вимог до візуального оформлення, включно з контрастністю, кольоровою схемою та взаємним розташуванням кнопок. Врахування цих вимог є необхідним для побудови інтерфейсу, орієнтованого на точність та зручність.

Завдання розроблення системи передбачає також побудову моделі взаємодії, у якій погляд визначає послідовність команд. Така модель повинна охоплювати початкове виявлення погляду, фільтрацію сигналу, аналіз фіксацій, активацію символів, а також ведення користувацьких даних. Усі ці елементи повинні працювати узгоджено, формуючи завершений механізм введення тексту [9].

Загалом постановка задачі у цьому дослідженні полягає у створенні програмної системи віртуальної клавіатури, що забезпечує введення тексту виключно через рух очей, з урахуванням технічних, фізіологічних та ергономічних особливостей користувача. Опис об’єкта дослідження демонструє комплексність взаємодії між людиною та системою, а також визначає параметри, що впливають на точність та стабільність введення. Реалізація такої системи повинна забезпечити доступність цифрових середовищ для людей з обмеженою фізичною рухливістю.

Для відображення загальної взаємодії користувача із системою доцільно представити модель у вигляді діаграми прецедентів. Вона демонструє основні дії, які користувач може виконувати під час роботи з віртуальною клавіатурою, керованою поглядом, та межі відповідальності системи у цих процесах. На рисунку 2.1 наведено схему роботи пристрою, що узагальнює функціональні можливості системи.

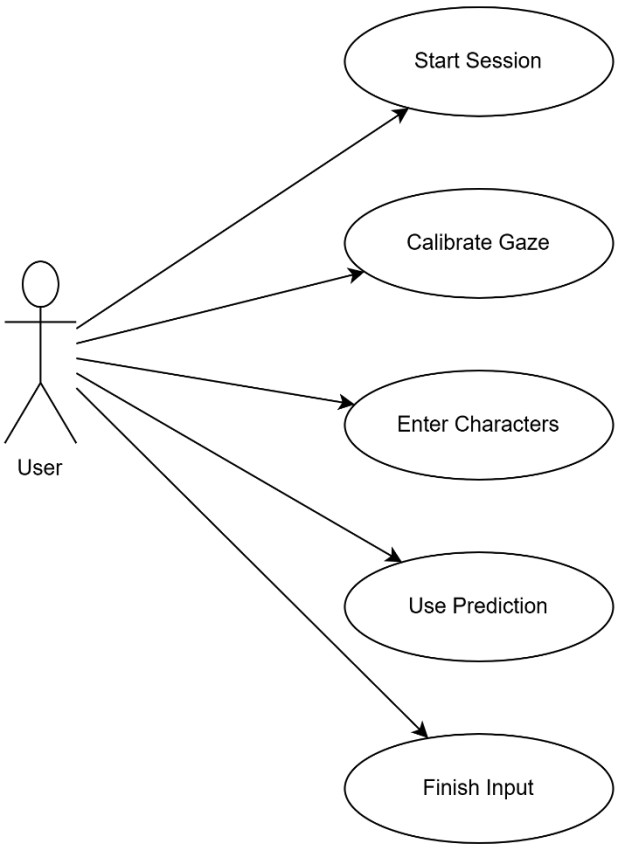

Рисунок 2.1 – Схему роботи пристрою користувацької взаємодії з системою введення тексту за поглядом

Як видно з схеми, користувач взаємодіє з системою через п’ять основних прецедентів: запуск сеансу, калібрування погляду, введення символів, використання прогнозування та завершення введення. Кожен прецедент відповідає окремому етапу використання системи і визначає набір дій, які система повинна підтримувати [10]. Діаграма підкреслює, що користувач є єдиним актором, а всі функції спрямовані на забезпечення комфортного та безпомилкового введення тексту за допомогою руху очей. Таким чином, схема є основою для подальшого моделювання внутрішніх процесів у підрозділі 2.1.

Структуру програмних компонентів, які реалізують обробку погляду та забезпечують взаємодію користувача з віртуальною клавіатурою, доцільно подати у вигляді діаграми класів. Вона дозволяє показати, як окремі модулі системи пов’язані між собою та які функції виконує кожен з них. На рисунку 2.2 представлено UML

Class Diagram, що відображає основні класи, відповідальні за обробку координат погляду, визначення фіксацій та вибір символів у процесі введення тексту.

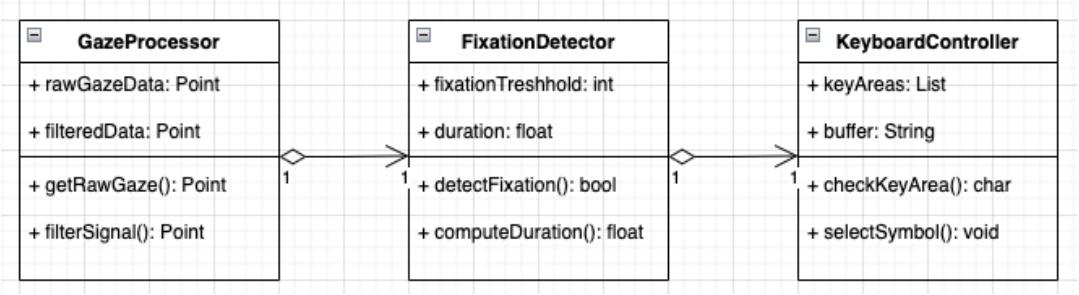

Рисунок 2.2 – UML Class Diagram основних програмних модулів системи введення тексту за поглядом

На діаграмі наведено три ключові класи, що формують основу програмної архітектури системи. Клас *GazeProcessor* забезпечує отримання та попередню фільтрацію координат погляду, що надходять з трекера. Клас *FixationDetector* відповідає за аналіз стабільності цих координат та визначення фіксацій, необхідних для подальшої інтерпретації взаємодії користувача з інтерфейсом. Клас *KeyboardController* здійснює зіставлення фіксацій із координатами клавіш, виконує логіку вибору символу та додає його в текстовий буфер. Така структуризація модулів дозволяє чітко розмежувати відповідальність компонентів системи й забезпечує можливість масштабування функціональності у разі подальшого розвитку проєкту.

2.2. Інформаційна модель системи введення за допомогою погляду

Інформаційна модель системи відображає логіку перетворення руху очей у керувальні дії, необхідні для роботи віртуальної клавіатури. Вона описує складові, що формують потоки даних, а також взаємозв’язки між етапами отримання, обробки та інтерпретації координат погляду. Така модель дозволяє встановити структуру даних, що є основою для алгоритмів вибору символів та загальної роботи інтерфейсу.

Першою сутністю, яка надходить до системи, є сирі координати погляду. Це дані, що генеруються трекером у режимі реального часу й містять інформацію про положення очей на площині екрана. Кожен запис містить координати та часову мітку,

що забезпечує можливість аналізу тривалості та стабільності фіксацій. Структура цієї сутності подана в Таблиці 2.1.

Таблиця 2.1 – Структура сутності RawGazePoint

Поле	Тип	Опис
X	float	Горизонтальна координата погляду
Y	float	Вертикальна координата погляду
Timestamp	float	Час реєстрації точки

На основі послідовності таких координат система формує фіксації – ділянки, на яких погляд користувача утримується достатньо стабільно, щоб виконати вибір символу. Фіксації визначаються алгоритмами згладжування та порогової тривалості. Вони є ключовими для визначення того, чи справді користувач намагається вибрати елемент інтерфейсу, оскільки погляд природно коливається навіть під час концентрації. Структура фіксації наведена в Таблиці 2.2.

Таблиця 2.2 – Структура сутності Fixation

Поле	Тип	Опис
CenterX	float	Центр фіксації по осі X
CenterY	float	Центр фіксації по осі Y
Duration	int	Тривалість утримання погляду (мс)

Віртуальна клавіатура представлена набором клавіш, кожна з яких має координатні межі. Саме порівняння координат фіксації з межами клавіші визначає факт взаємодії. Інформаційна модель описує цей процес як послідовність перевірок перетину області фіксації та області клавіші. У результаті формується подія вибору, яка містить інформацію про символ та час активації.

Потік даних у моделі організовано у вигляді етапів: отримання координат, фільтрація, виявлення фіксацій, визначення натискання, формування тексту. Кожен етап працює зі своїм набором даних, але всі вони взаємопов’язані: зміна на одному рівні впливає на кінцевий результат. Саме тому структура даних повинна бути узгодженою й формалізованою [11].

Калібрування є важливою частиною інформаційної моделі, оскільки воно дозволяє пов’язати координати трекера з реальною позицією елементів інтерфейсу. У

процесі калібрування система збирає кілька орієнтовних точок, за якими формує перетворення між простором камери та простором екрана. Надалі всі координати коригуються відповідно до цих значень.

Модель також передбачає можливість зберігання статистичних даних про введення: кількість виборів, помилкові активації, середню тривалість фіксацій і швидкість введення. Це дозволяє аналізувати ефективність системи та налаштовувати інтерфейс відповідно до індивідуальних потреб користувача.

У поєднанні всі описані сутності формують структуру даних, яка дозволяє системі коректно інтерпретувати рух очей і перетворювати його у текст. Інформаційна модель є основою для розробки алгоритмів, приведення інтерфейсу до стандартизованої форми та забезпечення надійності взаємодії. Використання лише необхідних сутностей робить модель компактною, але достатньо повною для реалізації функціональної системи введення тексту через погляд. Узагальнення описаних сутностей та їх взаємодії дає змогу сформувати структурну архітектуру роботи системи [12]. Оскільки процес перетворення руху очей у текстове введення складається з послідовних етапів обробки, доцільно представити його у вигляді блок-схеми. Це дозволяє наочно продемонструвати, як дані переміщуються між модулями, та підкреслює логічну послідовність роботи системи. Загальну архітектуру подано на рис. 2.3.

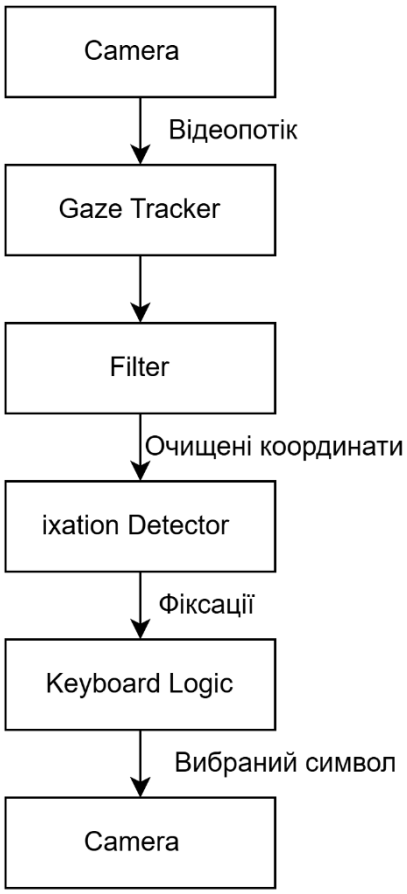

Рисунок 2.3 – Схема архітектури системи введення тексту за допомогою погляду

Схема демонструє поетапність обробки даних: від отримання відеопотоку камерою до формування текстового результату. Перші три етапи пов’язані з аналізом і стабілізацією координат погляду, що забезпечує їх придатність для подальшої інтерпретації. Наступний етап визначення фіксацій, встановлює моменти, коли погляд зупиняється в межах конкретної клавіші. Завершальні модулі відповідають за логіку вибору символу та формування текстового рядка.

2.3. Структура вхідних та вихідних даних системи

Вхідні дані системи введення тексту за допомогою погляду формуються з інформації, яку генерує пристрій відстеження. Основу цих даних складають координати погляду, що надходять у вигляді потоків точок, кожна з яких містить

горизонтальну і вертикальну позицію на площині екрана. Ці дані є первинним джерелом для побудови подальших рівнів обробки, тому їх точність визначає ефективність роботи всієї системи. Окрім координат, на вхід надходять часові мітки, що супроводжують кожну точку руху очей. Вони дозволяють визначати тривалість фіксацій, швидкість переміщення погляду та переходи між різними зонами інтерфейсу. Часові параметри є критичними для відділення фіксацій від саккад і формують основу алгоритмів вибору символів у віртуальній клавіатурі.

До вхідних даних належать також показники впевненості пристрою у коректності визначення координат. Це можуть бути внутрішні значення трекера погляду, що сигналізують про наявність шумів або часткову втрату ока у кадрі. Використання таких показників дає змогу фільтрувати ненадійні точки та запобігати помилкам під час обирання елементів інтерфейсу. Ще однією групою вхідних даних є параметри освітлення або індикатори якості зображення, які надсилаються деякими моделями трекерів. Ці метадані дозволяють системі визначати, чи не погіршився стан зовнішніх умов, і відповідно коригувати чутливість або розмір кнопок на клавіатурі. Таким чином, система отримує можливість адаптуватися до різних ситуацій без участі користувача [13].

Дані калібрування також входять до переліку вхідних. Вони включають координати точок, на яких користувач фокусував погляд під час початкового налаштування, а також коефіцієнти трансформації, що узгоджують реальні координати з логічними зонами інтерфейсу. Ці дані використовуються на всіх етапах обробки й забезпечують точне позиціонування погляду на клавішах.

До важливих категорій вхідних даних належать параметри інтерфейсу, такі як розмір кнопок, їхній колір, відстань між елементами та режим масштабування. Хоча вони не надходять від користувача у прямому вигляді під час роботи, вони зберігаються в системі та впливають на спосіб інтерпретації координат погляду. Це дозволяє коректно визначати подію вибору символу. У системах з інтегрованими алгоритмами прогнозування тексту вхідні дані включають також статистику попередніх введень, частоту використання слів і внутрішні словники. Такі дані використовуються для побудови рекомендацій, що скорочують кількість фіксацій,

необхідних для введення фраз. Статистика постійно оновлюється у процесі роботи системи, формуючи індивідуальний профіль. До вхідних даних відносять також стан користувача, який може оцінюватися за непрямими параметрами: швидкістю руху очей, кількістю помилкових спроб вибору або зміною тривалості фіксацій. Ці значення допомагають інтерфейсу адаптуватися під фізичний чи емоційний стан користувача, забезпечуючи плавність роботи в умовах втоми або напруження.

Вихідні дані системи формуються після багаторівневої обробки погляду та відповідають символам, які були вибрані користувачем. Основним вихідним елементом є текстовий рядок, що поповнюється при кожній події активації клавіші. Текст може передаватися у зовнішні програми, вікна чатів, документи або інші середовища, що використовує користувач.

До вихідних даних системи належать також повідомлення про події, які фіксують вибір конкретних кнопок, завершення слова, прийняття прогнозованого варіанту або очищення тексту. Такі повідомлення можуть використовуватися у журналах дій, що дозволяє проводити подальший аналіз і вдосконалювати роботу системи відповідно до стилю користувача. Важливою групою вихідних даних є внутрішні сигнали стану системи: підтвердження вибору символу, успішність прогнозування або індикація помилки у фіксації. Вони передаються між модулями програми й визначають, які етапи взаємодії потрібно виконати далі [14]. Це забезпечує узгодженість роботи всіх компонентів системи. Вихідними можуть бути також адаптивні зміни інтерфейсу, що застосовуються після аналізу поведінки користувача. Наприклад, система може автоматично збільшити розмір кнопок, змінити інтервал активації або запропонувати спрощену розкладку. У таких випадках вихідні дані включають параметри оновленого інтерфейсу.

У разі використання статистичного блоку вихідні дані можуть включати звіти про продуктивність: середню швидкість введення, кількість помилкових виборів, час фіксацій або динаміку зміни точності. Такі дані цінні для реабілітаційних центрів, де система може використовуватися як інструмент оцінки прогресу користувача. Окремим видом вихідних даних є підказки, сформовані модулем прогнозування. Вони містять набір слів або фраз, які система пропонує користувачу на основі

попереднього введення. Поява таких підказок зменшує навантаження на користувача і дозволяє прискорити введення тексту. Структура вхідних та вихідних даних охоплює широкий спектр елементів від координат погляду до адаптивних параметрів інтерфейсу. Узгоджена робота цих даних визначає ефективність системи введення та її здатність забезпечувати точний і комфортний механізм набору тексту для людей з обмеженою рухливістю. Продумана організація інформаційних потоків є основою для подальшого проектування програмної логіки та архітектури системи.

Динаміку передачі інформації між модулями системи доцільно відобразити у вигляді діаграми послідовності. Така діаграма показує, які компоненти системи беруть участь у формуванні вихідних даних, у якій послідовності здійснюється обробка сигналів погляду та як формується кінцевий результат у вигляді вибраного символу. На рисунку 2.4 наведено UML Sequence Diagram, що відображає послідовний обмін повідомленнями між ключовими компонентами системи.

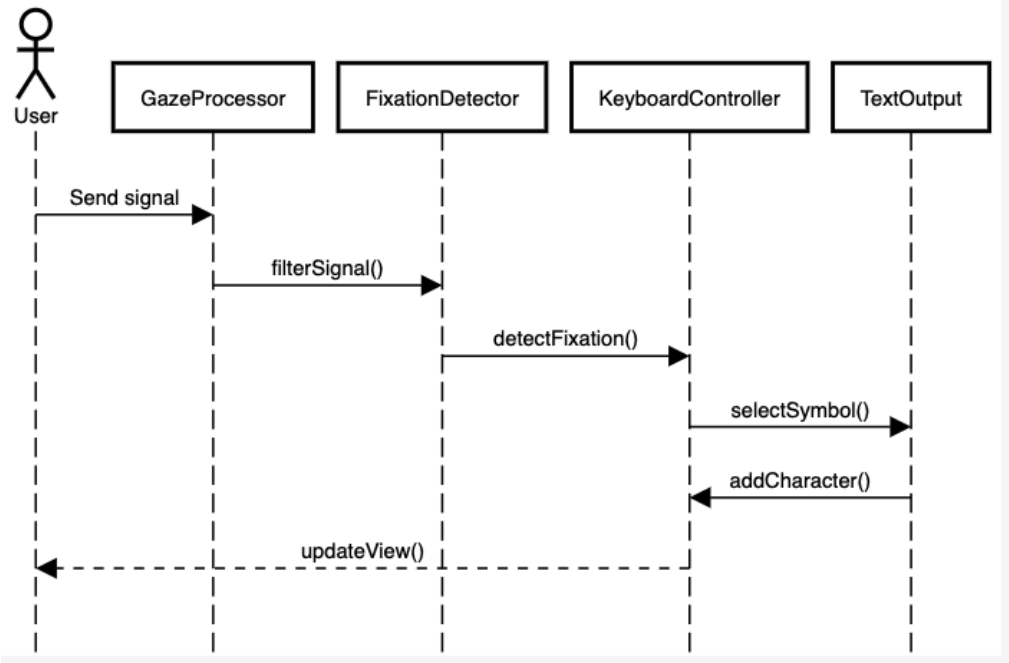

Рисунок 2.4 – UML Sequence Diagram процесу обробки погляду та формування символу

Як видно з діаграми, обробка даних починається з отримання координат погляду модулем GazeProcessor. Після фільтрації сигнал передається до модуля FixationDetector, який визначає наявність фіксації. Далі фіксація надходить до KeyboardController, де відбувається перевірка відповідності погляду певній клавіші та

формування вибраного символу. Модуль TextOutput додає символ до текстового рядка та повертає підтвердження назад контролеру. Така послідовність взаємодій демонструє, як саме формуються вихідні дані системи та в якій структурі вони надходять до текстового блоку.

2.4. Вимоги до системи та аналіз обмежень

Вимоги до системи введення тексту за допомогою погляду визначають набір характеристик, які необхідні для забезпечення точності, надійності та зручності роботи користувача. Оскільки інтерфейс орієнтований на людей з обмеженою рухливістю, система повинна працювати стабільно у різних умовах, без потреби у складних маніпуляціях або ручному налаштуванні. Це визначає високий рівень автоматизації процесів і адаптивність алгоритмів до особливостей поведінки користувача.

Однією з головних вимог є точність визначення координат погляду. Система повинна коректно розпізнавати фіксації навіть за умов мікрорухів очей, зміни положення голови або часткової втрати контакту з камерою. Висока точність є основою для мінімізації хибних натискань, що особливо важливо для користувачів, які не можуть швидко виправити помилки через фізичні обмеження. Вимога стабільності роботи охоплює здатність системи функціонувати без раптових збоїв, затримок або зниження продуктивності. Інтерфейс введення тексту повинен реагувати у реальному часі, оскільки навіть незначні затримки можуть призвести до втрати контролю над введенням [15]. Це визначає необхідність оптимізованих алгоритмів обробки відеопотоку та подій.

Система повинна підтримувати адаптивність інтерфейсу, тобто можливість змінювати розмір, контрастність і розташування клавіш відповідно до потреб користувача. Люди з різними формами моторних порушень мають відмінні можливості контролю погляду, тому універсальний набір параметрів не може бути ефективним для всіх. Адаптивність дозволяє зменшувати навантаження на зір та збільшувати точність введення.

До функціональних вимог належить підтримка прогнозування тексту, яке повинно зменшувати кількість необхідних фіксацій. Це підвищує швидкість введення та знижує втому користувача. Водночас система має підтримувати можливість вимкнення або налаштування прогнозування, оскільки не всі користувачі однаково сприймають роботу таких алгоритмів. Важливою вимогою є наявність механізму калібрування, що забезпечує адаптацію системи до індивідуальних параметрів очей. Калібрування повинно виконуватися швидко та без складних інструкцій, оскільки користувач може не мати змоги самостійно виконувати точні рухи. Система має автоматично пропонувати повторне калібрування у разі погіршення точності.

До нефункціональних вимог належить вимога сумісності з різними моделями трекерів погляду або вебкамер. Це забезпечує доступність системи та її можливість працювати на різних пристроях без необхідності використання дорогого спеціалізованого обладнання. Сумісність підвищує практичну цінність системи у домашніх і реабілітаційних умовах. Система повинна підтримувати низький рівень навантаження на апаратні ресурси. Оскільки вона може запускатися на побутових комп'ютерах або планшетах, важливо забезпечити мінімальне використання процесорного часу та пам'яті. Оптимізація обробки зображень та подій дозволяє уникнути перегріву пристрою та забезпечує стабільність роботи протягом тривалих сеансів [16]. Окремою вимогою є безпечна робота системи. Вона має гарантувати, що введення тексту виконуватиметься лише за умов реальної фіксації погляду, а не через випадкові шуми або відблиски. Система повинна запобігати неконтрольованому вибору символів, що може призвести до помилок у критичних ситуаціях, наприклад, під час спілкування чи введення медичної інформації.

Аналіз обмежень показує, що найсерйозніші з них стосуються фізіологічних особливостей руху очей. Користувач не може утримувати погляд у цілком нерухомому стані, тому будь-яка система має враховувати природну нестабільність та стратегії компенсації. Це обмеження визначає необхідність складних алгоритмів фільтрації даних.

Серед технічних обмежень ключовим є залежність від якості камери або трекера погляду. Недостатня роздільна здатність або низька частота зчитування

можуть різко зменшити точність визначення координат. Це обмеження накладає вимогу використання мінімальних апаратних параметрів, які повинні бути дотримані для стабільної роботи системи. Обмеження, пов’язані з умовами освітлення, також впливають на ефективність системи. Надмірні відблиски, тіні або різкі зміни освітлення можуть призвести до втрати ока у кадрі. Тому система повинна мати механізми визначення невідповідних умов та автоматичної компенсації, хоча повністю усунути вплив освітлення неможливо.

Ще одним обмеженням є швидкість введення тексту, яка залишається нижчою порівняно з традиційними методами. Навіть за умови використання прогнозування система не здатна досягти швидкості набору на фізичній клавіатурі. Це обмеження визначає необхідність оптимізації інтерфейсу, щоб мінімізувати кількість фіксацій [17]. Загалом вимоги та обмеження системи формують рамки, у яких може працювати віртуальна клавіатура, керована поглядом. Вони визначають параметри точності, адаптивності та стабільності, необхідні для забезпечення комфортної взаємодії користувача з інтерфейсом. Аналіз обмежень дозволяє сформувати реалістичні технічні рішення та прогнозувати поведінку системи в різних умовах експлуатації.

Висновки до розділу

Сформовані вимоги до системи демонструють, що віртуальна клавіатура, керована поглядом, повинна забезпечувати високу точність визначення координат, стабільність роботи та адаптивність інтерфейсу до різних категорій користувачів. Реалізація цих вимог є основою для ефективної взаємодії та зменшення кількості помилкових виборів.

Аналіз обмежень показав, що на роботу системи суттєво впливають фізіологічні особливості руху очей, технічні характеристики камер і умови освітлення. Ці фактори формують межі, у яких система може функціонувати, та визначають потребу у застосуванні алгоритмів фільтрації, компенсації шумів і адаптивного регулювання параметрів.

Отже, врахування вимог і обмежень дозволяє створити реалістичну архітектуру системи та визначити ключові напрями оптимізації. Це забезпечує основу для подальшої розробки математичних моделей, алгоритмів і програмних рішень, що відповідатимуть потребам людей з обмеженою рухливістю.

РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Методи обробки координат погляду

Обробка координат погляду визначає якість роботи всієї системи введення тексту, оскільки саме ці дані формують основу для розпізнавання вибору користувача. Трекер погляду передає сирий сигнал, який містить координати зіниці або відблиску на рогівці, отримані з відеокадрів камери. Цей сигнал нерівномірний, тому потребує багатоетапної обробки. Першочерговим завданням є побудова стабільної траєкторії руху очей, яка відображатиме наміри користувача, а не випадкові зміщення, спричинені природною мінливістю зорової системи.

Перший етап полягає в очищенні сирих даних від шумів. Система повинна компенсувати дрібні коливання погляду, які не мають стосунку до взаємодії з інтерфейсом. Для цього застосовуються прості методи згладжування середнє ковзне, медіанний фільтр або експоненційне згладжування. Вони дозволяють утворити більш рівномірну траєкторію, що підвищує точність подальших етапів. Проте такі методи не завжди достатні для роботи з користувачами, у яких рухи погляду можуть бути нестабільними через фізіологічні особливості. У складніших випадках використовуються моделі прогнозування, серед яких найбільш поширеним є фільтр Калмана [18].

Цей підхід дозволяє враховувати історію руху погляду та математично оцінювати майбутнє положення очей. Фільтр поєднує виміряні координати з передбачуваною траєкторією, що дає змогу зменшити вплив шумів. Це особливо помітно у ситуаціях, коли користувач змінює положення голови або камера фіксує часткові втрати зображення очей.

Наступним кроком є розділення рухів очей на саккади та фіксації. Саккади це швидкі переміщення, під час яких координати погляду змінюються стрибкоподібно та не можуть розглядатися як свідчення наміру вибору. Фіксації, навпаки, відображають стійку увагу на певній точці, що дозволяє інтерпретувати їх як потенційний вибір клавіші. Методи визначення фіксацій базуються на порогах

швидкості або дисперсії координат, що дозволяє виділити стабільні зони на траєкторії.

Одним з найбільш поширених методів є алгоритм I-VT (Identification by Velocity Threshold). Його суть полягає у розрахунку швидкості зміни координат погляду між послідовними точками та визначенні періодів, коли швидкість падає нижче заданого порога. У ці моменти система вважає, що користувач зупинив погляд на об’єкті. Цей метод є придатним для віртуальних клавіатур, оскільки він легко налаштовується під різні швидкісні характеристики користувача.

Іншим підходом є алгоритм I-DT (Identification by Dispersion Threshold). Він аналізує розкид координат у певному часовому вікні та визначає фіксацію тоді, коли цей розкид залишається малим. Такий метод добре працює у ситуаціях, коли користувач здатен утримувати погляд у відносно стабільному положенні, але погано реагує на коливання швидкості. Обидва підходи можуть поєднуватися, утворюючи комбіновані методи, що враховують як швидкість, так і дисперсію [19]. Після визначення фіксацій система повинна зіставити їх із координатами клавіш віртуальної клавіатури. Для цього використовується метод перевірки входження точки у прямокутну область кнопки. Якщо центр фіксації знаходиться в межах певної клавіші, система інтерпретує це як потенційний вибір. На цьому етапі важливо враховувати час утримання погляду, щоб уникнути випадкових активацій, спричинених мікрорухами очей.

Час фіксації є важливим параметром для визначення намірів користувача. Занадто коротка фіксація призводить до помилкових виборів, а надто довга знижує швидкість введення. Тому алгоритми формують оптимальний інтервал, який може адаптуватися під індивідуальний стиль користувача. Це дозволяє підвищити комфорт системи, оскільки користувачі з різним рівнем контролю погляду матимуть свої параметри часу вибору. Для підвищення точності системи використовують методи компенсації рухів голови. Навіть невеликий нахил або зміщення голови може змінити співвідношення між координатами камери та логічними координатами екрана. Алгоритми компенсують такі зміщення шляхом коригування матриці перетворення, яка отримується під час калібрування. Завдяки цьому координати погляду

залишаються точними навіть за нестандартного положення користувача. Особливу увагу приділяють методам фільтрації мікрорухів, що виникають у людей з обмеженою рухливістю. У таких користувачів рух очей може бути менш контрольованим, тому система повинна відрізняти реальні фіксації від хаотичних переміщень. Для цього застосовуються алгоритми, що враховують мінімальний радіус стабільної зони чи визначають фіксацію на основі кількох сусідніх точок.

Деякі системи використовують машинне навчання для класифікації рухів погляду. Моделі, натреновані на даних від різних користувачів, можуть визначати фіксації більш точно, ніж класичні порогові методи. Це стає актуальним, коли профіль рухів очей суттєво відрізняється від типового наприклад, у людей із неврологічними порушеннями, які мають нерівномірні або нестабільні рухи очей [20].

Методи обробки координат погляду також включають контроль якості вхідного сигналу. Це необхідно для того, щоб система могла виявляти моменти, коли камера не бачить око або рівень освітлення недостатній. У таких випадках система тимчасово призупиняє аналіз, щоб уникнути помилок у виборі символів. Подібний підхід підвищує надійність роботи в реальних умовах. Ще одним напрямом є оцінка траєкторій погляду для виявлення закономірностей, що можуть допомогти у прогнозуванні намірів користувача. Наприклад, рух погляду до краю клавіші часто свідчить про намір перейти до іншої кнопки. Використання таких закономірностей дозволяє створювати адаптивні інтерфейси, які реагують більш природно.

На завершальному етапі система формує сигнал вибору символу та передає його у модуль текстового виводу. Методи обробки координат погляду повинні забезпечити безпомилкове визначення моменту вибору, оскільки саме це визначає точність введення. Ефективне поєднання фільтрації, класифікації рухів та адаптації параметрів дозволяє створити систему, придатну для щоденного використання користувачами з різними формами порушень рухливості.

Узагальнюючи, методи обробки координат погляду охоплюють цілий спектр рішень від класичних фільтрів до моделей машинного навчання. Їхнє поєднання дає змогу створити надійну основу для віртуальної клавіатури, що реагує на погляд користувача та підтримує стабільний процес введення тексту.

3.2. Моделі визначення фіксацій та вибору символа

Моделі визначення фіксацій є центральною частиною системи введення тексту поглядом, оскільки саме вони формують основу для інтерпретації наміру користувача. Фіксація описує момент, коли координати погляду залишаються відносно стабільними протягом певного часу, що дозволяє системі розпізнати вибір тієї чи іншої клавіші. Через природну рухливість очей, яка включає мікросаккади, дрейф та короткі стрибки між точками, визначення фіксацій потребує застосування математичних критеріїв стабільності координат.

Однією з найпоширеніших моделей є I-VT (Identification by Velocity Threshold), яка базується на швидкості руху погляду. Її принцип полягає в тому, що фіксацією вважаються ті інтервали часу, де швидкість зміни координат не перевищує встановленого порогу. Це дозволяє відділяти повільні, стабільні рухи, характерні для зосередженого погляду, від швидких саккад. Формально швидкість визначається як:

$$\square = \frac{\sqrt{(\square_{\square} - \square_{\square-1})^2 + (\square_{\square} - \square_{\square-1})^2}}{\square_{\square} - \square_{\square-1}} \quad (3.1)$$

Погляд вважається фіксацією, якщо $v < V_{th}$, де V_{th} поріг швидкості, що задається експериментально.

Інша модель I-DT (Identification by Dispersion Threshold) ґрунтується не на швидкості, а на просторовій розбіжності точок. За цим підходом фіксацією вважається група точок, у яких максимальна різниця координат не перевищує заданий поріг. Метод визначає фіксацію, якщо:

$$max(x) - min(x) < D_x imax(y) - min(y) < D_y$$

Цей підхід добре працює у випадках, коли погляд коливається мінімально, але може давати помилки при наявності дрібних мікрорухів.

Для систем введення тексту поглядом доцільно поєднувати обидва критерії, оскільки користувачі з моторними порушеннями можуть демонструвати різні типи рухів очей. Комбінація моделей дозволяє зменшити кількість хибних виборів, підвищуючи стійкість алгоритму до зовнішніх та фізіологічних факторів. Зазвичай

використовується адаптивний підхід: якщо швидкісний критерій не дає чіткої відповіді, система застосовує аналіз розсіювання [21].

Порівняння цих моделей часто подається у табличному вигляді, що дозволяє оцінити придатність кожної з них у контексті побудови інтерфейсу віртуальної клавіатури. Узагальнений вигляд наведено нижче.

Таблиця 3.1 – Порівняння моделей визначення фіксацій

Модель	Критерій	Переваги	Недоліки
I-VT	Швидкість руху погляду	Висока стійкість до мікрорухів, проста реалізація	Чутливість до шумів та втрат кадру
I-DT	Просторова розсіювання	Добре виділяє стабільні ділянки	Гірше працює при дрейфі погляду
Комбінована	Швидкість + розсіювання	Найвища точність, адаптивність	Потребує калібрування параметрів

Вибір моделі впливає не лише на визначення фіксацій, але й на подальший етап вибір символа. Після визначення центру фіксації система повинна зіставити його з межами клавіш. Для цього використовується перевірка входження точки у прямокутну область клавіші. Якщо центр фіксації належить до області клавіші та тривалість фіксації перевищує мінімальний час вибору, система інтерпретує це як активацію.

Одним із практичних способів описати цей процес є використання порогової тривалості. Система аналізує, чи перевищує фіксація час T_{fix}, що дозволяє запобігти вибору символу через короткі випадкові погляди. Такі критерії особливо важливі для користувачів, які не завжди можуть контролювати дрібні коливання очей.

Для узагальнення послідовності виконання операцій доцільно подати процес визначення фіксації та вибору символа у вигляді UML Activity Diagram. На рисунку 3.1 наведено діаграму, що відображає покроковий алгоритм переходу від отримання координат погляду до формування вибраного символа.

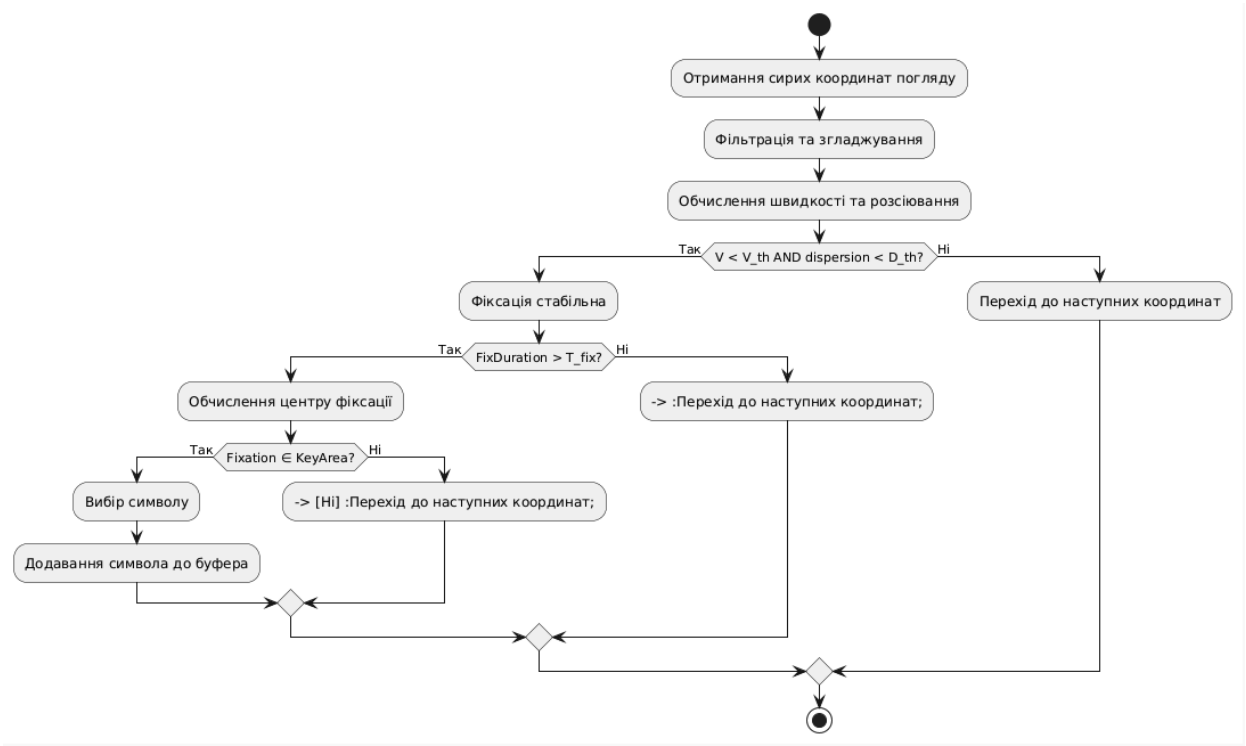

Рисунок 3.1 – UML Activity Diagram алгоритму визначення фіксації та вибору символа

У діаграмі послідовність обробки структурована у вигляді взаємопов’язаних етапів, що демонструють рух даних від моменту реєстрації координат до формування події вибору. Вона дозволяє наочно побачити, як система переходить між станами залежно від швидкості погляду, розсіювання координат, тривалості фіксації та збігу положення погляду з областю клавіші. Такий підхід підкреслює логіку прийняття рішень у системі та допомагає формалізувати вимоги до алгоритмів визначення фіксацій.

Для підвищення точності інтерфейс може застосовувати додаткове обмеження у вигляді зони толерантності: навіть якщо погляд коливається в межах невеликого діапазону, система все одно зараховує його вибір. Це дозволяє компенсувати ефекти тремору або незначного зміщення голови під час введення тексту [22].

У деяких системах використовується адаптивне збільшення розміру активної зони навколо клавіш. Це доречно для ситуацій, коли користувач демонструє нестабільні фіксації через втому або особливості зорово-моторної координації. Такий механізм дозволяє зменшити кількість помилкових відхилень і забезпечити більш

комфортний процес набору. Оскільки точність вибору символа залежить від того, як швидко система обробляє фіксації, у моделі вводиться обмеження на латентність обробки. Чим менше часових затримок між визначенням фіксації та активацією, тим природніше сприймається введення тексту. Затримки понад 100 мс можуть відчуватися користувачем як «пригальмовування» та зменшувати ефективність роботи.

Для уточнення логіки вибору символа доцільно представити коротку структурну модель у вигляді псевдокоду, що демонструє послідовність обробки фіксацій:

```
if fixation_detected:
    if fixation.duration > T_fix:
        key = keyboard.getKeyAt(fixation.center)
        if key is not None:
            output.append(key.symbol)
```

Рисунок 3.2 – псевдокод алгоритму послідовності обробки фіксацій

Такий алгоритм демонструє базову логіку прийняття рішень без прив’язки до конкретної реалізації, зберігаючи універсальність підходу.

Моделі визначення фіксацій також можуть взаємодіяти з алгоритмами прогнозування тексту. У цьому випадку система формує кілька пропозицій, і користувач може зафіксувати погляд на одному з них. Це значно скорочує кількість рухів, необхідних для введення довгих слів або фраз, що особливо корисно в умовах обмеженої рухливості. Після завершення етапу вибору символа система формує текстовий вихід, оновлює статистику введення та може адаптувати параметри інтерфейсу, якщо виявлено труднощі , наприклад, збільшити час фіксації або змінити розмір клавіш. Таким чином, моделі визначення фіксацій стають основою адаптивного механізму, що підлаштовується під поточний стан користувача.

Узагальнюючи, модель визначення фіксацій та вибору символа поєднує математичні критерії стабільності погляду, аналіз просторових координат і адаптивні механізми підвищення точності. Її коректне налаштування дозволяє перетворити

природний рух очей у надійний інструмент введення тексту, що є ключовим для ефективності всієї системи.

3.3. Алгоритми оптимізації розкладки віртуальної клавіатури

Оптимізація розкладки віртуальної клавіатури має вирішальне значення для підвищення ефективності введення тексту поглядом. Вона спрямована на скорочення кількості рухів очей, зниження кількості помилок та створення умов, у яких користувач може вводити текст без надмірного напруження. Для людей з обмеженою рухливістю це особливо суттєво, оскільки навіть незначні неточності у розташуванні клавіш можуть призвести до значних труднощів.

Багато підходів до оптимізації ґрунтуються на частотному аналізі символів. Основна ідея полягає в тому, щоб часто використовувані символи розташовувалися у найбільш доступних зонах інтерфейсу, де погляд стабілізується швидше. Це дозволяє зменшити середню траєкторію руху очей та підвищити швидкість введення. На відміну від фізичних клавіатур, віртуальні системи не мають конструктивних обмежень, тому розкладка може змінюватися динамічно [23].

У системах введення тексту поглядом важливим є автоматичне регулювання розміру клавіш. Клавіші, які користувач вводить найчастіше, можуть збільшуватися, щоб зменшити ризик помилкових виборів. Це може здійснюватися за допомогою простої адаптивної формули:

$$□_{□□□} = □_{□□□□} \cdot (1 + □ \cdot □) \quad (3.2)$$

Де

S_{new}— новий розмір клавіші,

S_{base}— базовий розмір,

E— частка помилкових виборів для цієї клавіші, k— коефіцієнт чутливості.

Таке регулювання дозволяє інтерфейсу автоматично адаптуватися до стилю роботи користувача та зменшує навантаження на зір.

Ще одним методом оптимізації є мінімізація відстані між клавішами, які часто використовуються послідовно. Для цього застосовуються алгоритми, що оцінюють частотність переходів між парами символів. Метою є мінімізація функції:

$$\square = \sum_{\square,\square} \square\square\square\square(\square,\square) \cdot \square\square\square\square(\square,\square) \quad (3.3)$$

Де

$freq(i,j)$— частота появи пари символів $i \to j$,

$dist(i,j)$— відстань між клавішами.

Цей підхід зменшує час переміщення погляду між символами та робить введення плавнішим.

Практичним інструментом для побудови оптимальної розкладки є аналіз частот українських та англійських символів. Орієнтовні значення подано у таблиці 3.2.

Таблиця 3.2 – Частотність символів та рекомендації щодо оптимізації

Символ	Частота (%)	Рекомендоване розташування
О	9.28	Центр, велика клавіша
А	7.21	Центр ліворуч
Н	6.35	Верхня центральна зона
І	5.92	Права частина
Е	5.55	Верхня центральна зона

Такі дані дозволяють формувати базову оптимізацію розкладки для широкого кола користувачів.

Іншим підходом є побудова теплової карти (heatmap) погляду, яка відображає зони найстабільнішого фокусування. На практиці це означає, що система може визначити ділянки, де погляд утримується довше, та розмістити ключові символи саме там. Використання heatmap значно підвищує точність введення та знижує кількість фіксацій, які не призводять до вибору символа. У разі нестабільного контролю рухів очей оптимізація може включати збільшення активних зон навколо клавіш, розширення меж клавіш або застосування «гнучкої розкладки», де символ може зсуватися в область, де користувач стабільніше утримує погляд. Такі методи

особливо корисні для людей із хронічною втомою або порушенням контролю м’язів очей.

Окреме значення мають алгоритми адаптації розкладки до індивідуальних особливостей користувача. Система може аналізувати статистику введення: середню швидкість, кількість помилкових виборів, середню тривалість фіксацій. Якщо система визначає, що певна клавіша часто помилково активується, вона може автоматично збільшити її розмір або змінити положення. У деяких сценаріях ефективним є застосування ієрархічних розкладок [24]. Користувач спочатку обирає групу символів (напр., голосні), після чого отримує збільшений набір клавіш усередині групи. Це особливо корисно для мов із великим алфавітом або для користувачів, які мають труднощі з точним фокусуванням.

Розкладки можуть оптимізуватися з урахуванням мови введення. Для англійської, української та інших мов використовуються різні частотні та статистичні моделі, що впливають на положення клавіш. Наприклад, в українській частотність голосних вища, що вимагає зміщення їх до центральної частини клавіатури.

Прогнозування тексту також впливає на оптимізацію розкладки. Якщо система формує підказки, вони повинні розміщуватися у найбільш доступних зонах. Для цього можна використовувати комбіновану функцію:

$$□□□□□ = □ \cdot □(□□□□) + □ \cdot □□□□(□□□□, □□□) \quad (3.4)$$

Це дозволяє розташовувати підказки з урахуванням того, наскільки близько до них утримується погляд.

Ще одним напрямом є оптимізація форми клавіш. Округлені або скошені форми можуть зменшувати ймовірність того, що погляд «зависне» у зоні між клавішами. Оптимізація форми є менш поширеною, але може суттєво покращити досвід користувачів з нестійкими фіксаціями. Зміна масштабу клавіатури під час введення може використовуватися як динамічна оптимізація. Наприклад, система може збільшувати розмір поруч розташованих клавіш одразу після вибору першої, щоб полегшити наступний вибір.

Потужним інструментом оптимізації є алгоритми машинного навчання. Вони аналізують повну історію введень, помилки, швидкість і навіть характер рухів очей. На основі цих даних система формує унікальну розкладку, яка з часом може суттєво відрізнятися від початкової. Це забезпечує максимально персоналізований досвід.

Для уточнення роботи оптимізаційних алгоритмів можна представити короткий псевдокод, що описує логіку адаптації розкладки:

```
if ErrorRate(key) > threshold:
    increaseSize(key)
if SlowFixations():
    shiftKeysToStableZones()
if PredictiveModelSuggests(word):
    movePredictionToCentralArea()
```

Рисунок 3.3 – псевдокод алгоритму адаптації розкладки

Це демонструє базові принципи адаптивної перебудови клавіатури.
Оптимізація розкладки віртуальної клавіатури поєднує частотний аналіз, аналіз траєкторій погляду, адаптивні моделі та алгоритми машинного навчання. Правильно налаштована розкладка істотно підвищує точність введення тексту та знижує втомлюваність користувача. Завдяки цьому системи введення поглядом можуть стати зручними навіть для людей із тяжкими порушеннями рухливості.

3.4. Оцінювання точності та стабільності роботи моделі

Оцінювання ефективності системи введення тексту поглядом базується на аналізі метрик точності, стабільності та швидкодії алгоритмів розпізнавання фіксацій. Таке оцінювання дає змогу визначити, наскільки коректно модель інтерпретує наміри користувача та чи може вона забезпечити комфортне введення тексту у реальних умовах. Через те, що рух очей є природно нестабільним, вимірювання продуктивності має спиратися на кількісні показники, які точно відображають роботу системи.

Однією з основних метрик є точність вибору символів. Вона визначає частку коректно активованих клавіш відносно загальної кількості введених символів. Формула для точності має вигляд:

$$\square\square\square\square\square\square\square\square = \frac{\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square}{\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square} \cdot 100\% \quad (3.5)$$

Приклад розрахунку: якщо користувач зробив 120 виборів, із яких 108 були правильними, тоді

$$Accuracy = \frac{108}{120} \cdot 100\% = 90\%.$$

Це дозволяє оцінити, чи відповідає система вимогам до комфортної роботи.

Метрика помилкових виборів обчислюється як

$$\square\square\square\square\square\square\square\square\square\square = \frac{\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square}{\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square\square} \quad (3.6)$$

Якщо з тих самих 120 виборів 12 були помилковими, тоді

$$ErrorRate = \frac{12}{120} = 0.1(10\%).$$

Зростання ErrorRate свідчить про те, що система недостатньо точно розпізнає стабільні фіксації або має проблеми з розкладкою клавіатури.

Стабільність фіксації оцінюється за допомогою індексу FSI (Fixation Stability Index):

$$FSI = \frac{1}{n}\sum_{i=1}^{n} \sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2}, \quad (3.7)$$

де n— кількість координат у фіксації. Для прикладу: нехай фіксація містить 4 точки, а середнє положення $(\bar{x}, \bar{y}) = (500,300)$. Для набору точок (505,302), (498,297), (501,304), (503,299) середнє відхилення становитиме:

$$FSI \approx \frac{1}{4}(5.38 + 3.60 + 4.47 + 3.16) = 4.15.$$

Низьке значення FSI означає стабільний погляд, що підвищує ймовірність точного вибору символа. Зведення основних метрик подано в таблиці 3.2, що дозволяє систематизувати критерії оцінювання продуктивності моделі.

Таблиця 3.3 – Метрики оцінювання точності та стабільності моделі

Метрика	Формула	Опис
Accuracy	$\frac{CorrectSelections}{TotalSelections} \cdot 100\%$	Частка правильних виборів
Error Rate	$\frac{WrongSelections}{TotalSelections}$	Частка помилкових виборів
FSI	$\frac{1}{n}\sum\sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2}$	Стабільність фіксації
Latency	—	Час від фіксації до вибору
Selection Time	—	Тривалість вибору символу

Після обчислення основних метрик система дозволяє проаналізувати придатність моделі до щоденного використання. Наприклад, якщо Accuracy перевищує 90%, а ErrorRate не більше 10%, інтерфейс працює достатньо надійно. Натомість, коли FSI перевищує поріг, система може автоматично збільшувати розмір клавіш або підвищувати час фіксації [25].

Практичну частину оцінювання проводять шляхом тестового введення фраз. Наприклад, користувачу запропоновано набрати речення «Hello world». Реальні результати можуть містити 11 правильних і 2 неправильні вибори. У такому разі:

$$Accuracy = \frac{11}{13} \cdot 100\% = 84.6\%, ErrorRate = \frac{2}{13} = 15.3\% \qquad (3.8)$$

Таке співвідношення свідчить про необхідність оптимізації розкладки або зниження порогу стабільності. Важливою характеристикою є латентність , час, що проходить між визначенням фіксації та появою символа. Для комфортної роботи вона повинна бути меншою за 100–120 мс. Якщо у тестовому середовищі середня latency становить 85 мс, система працює плавно; якщо ж значення перевищує 150 мс, користувач може відчути затримку. Окремо оцінюють Selection Time середній час вибору символу. Наприклад, якщо для введення 20 символів користувач витратив 28 секунд, середній час становить:

$$SelectionTime = \frac{28}{20} = 1.4 \square.$$

Це прийнятний показник, але значення понад 2 секунди свідчать про перевантаженість інтерфейсу або складність контролю поглядом. Проведення серії експериментів дає змогу побудувати індивідуальний профіль користувача. Наприклад, зниження Accuracy наприкінці сеансу може свідчити про втому, що підкреслює необхідність адаптивної зміни часу фіксації.

Оцінювання також може включати побудову графіків зміни FSI та ErrorRate у часі. Якщо протягом 10-хвилинної сесії FSI зростає з 3.8 до 7.5, система повинна перейти у «режим підвищеної толерантності».

У підсумку результати всіх розрахунків формуються у звіт, що містить таблиці, значення метрик і рекомендації щодо оптимізації. Такий підхід забезпечує можливість об’єктивно порівнювати різні конфігурації розкладки та параметри фільтрації. Узагальнюючи, інтеграція точнісних метрик, прикладів розрахунків і експериментальних результатів дозволяє комплексно оцінити продуктивність моделі. Це формує підґрунтя для адаптації системи під конкретного користувача та вдосконалення алгоритмів розпізнавання фіксацій.

Висновки до розділу

У розділі розглянуто повний цикл обробки координат погляду від очищення сирих даних до формування вибору символа на віртуальній клавіатурі. Показано, що якість і стабільність цих даних визначають точність усієї системи введення тексту.

Проаналізовані моделі визначення фіксацій, зокрема I-VT, I-DT та комбіновані підходи, які дозволяють компенсувати шум, мікрорухи та варіативність поведінки користувачів. Саме їх правильне налаштування забезпечує надійне розпізнавання намірів під час роботи з інтерфейсом. Окрему увагу приділено алгоритмам оптимізації розкладки клавіатури, які зменшують кількість рухів очей і підвищують швидкість введення. Використання частотних моделей, heatmap-фіксацій та адаптивних змін розміру клавіш робить систему доступнішою для користувачів з різним рівнем контролю погляду.

Оцінювання точності та стабільності моделі за допомогою метрик Accuracy, Error Rate, FSI, Latency і Selection Time дозволяє об’єктивно визначати ефективність алгоритмів. Практичні приклади розрахунків демонструють можливість кількісно оцінювати роботу системи та коригувати її параметри.

Узагальнюючи, можна стверджувати, що поєднання фільтрації координат, адаптивних моделей фіксації, оптимізованої розкладки та точнісних метрик формує надійну основу для системи введення тексту поглядом. Такий підхід забезпечує точність, стабільність та індивідуальне налаштування, необхідне для комфортного використання технології.

РОЗДІЛ 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.1. Вибір технологій та архітектури програмної реалізації

Реалізація системи введення тексту за допомогою погляду потребує ретельного вибору технологій, здатних забезпечити стабільну обробку відеопотоку, визначення координат очей та формування подій вибору символів у режимі реального часу. Python було обрано як основну мову розроблення через наявність широкої екосистеми бібліотек для комп'ютерного зору, машинного навчання та прототипування інтерфейсів, а також завдяки її високій читабельності та простоті масштабування. Мова дозволяє швидко створювати функціональні модулі й інтегрувати їх у єдину систему. У проєкті використовується OpenCV, оскільки вона забезпечує ефективні алгоритми детекції очей, фільтрації зображення та визначення ключових точок. OpenCV підтримує роботу з потоковим відео та має оптимізовані функції, що дозволяють виконувати обчислення з мінімальною затримкою [26]. Завдяки цьому можливо реалізувати стабільний механізм визначення координат погляду. Для моделювання руху очей та згладжування шумів доцільно застосовувати фільтр Калмана або його спрощені аналоги. Python має відповідні модулі, такі як filterpy, які дозволяють інтегрувати математичні моделі прогнозування у загальний конвеєр обробки. Застосування таких фільтрів суттєво підвищує точність вибору символів та компенсує мікрорухи очей або втрати кадрів.

Визначення фіксацій реалізується за допомогою спеціалізованих алгоритмів, що використовують пороги швидкості або дисперсії координат. Python забезпечує можливість реалізації цих моделей без значних обчислювальних витрат, оскільки працює з ефективними структурами даних та матричними операціями через бібліотеку NumPy. Використання NumPy дозволяє виконувати розрахунки векторизовано, що підвищує продуктивність системи.

Формування віртуальної клавіатури та її адаптивної логіки реалізується у вигляді окремого модуля. Для розроблення інтерфейсу може використовуватися PyQt або Tkinter, залежно від вимог до візуального відображення та реактивності. PyQt дозволяє створити масштабовані графічні елементи з підтримкою анімацій та

динамічної зміни розміру кнопок, що узгоджується з адаптивними механізмами клавіатури.

Архітектура програмної системи будується за модульним принципом. Основні модулі включають: модуль відеозахоплення, модуль аналізу зображення, модуль визначення фіксацій, модуль обробки вибору символа, модуль інтерфейсу та модуль журналювання. Така структура забезпечує незалежність компонентів та можливість їхнього тестування, модернізації або заміни без зміни усієї системи. Передавання даних між модулями здійснюється через внутрішні події та структуровані об’єкти, що дозволяє уникнути надлишкових залежностей між компонентами. Подібний підхід забезпечує гнучкість архітектури та полегшує інтеграцію нових алгоритмів. Наприклад, модуль детекції погляду може бути замінений більш точною моделлю без змін у модулі вибору символа.

Система використовує архітектурний підхід pipeline, у межах якого обробка даних відбувається у вигляді послідовності етапів. До таких етапів належать:

1. відеозахоплення,
2. попередня обробка зображення,
3. визначення координат погляду,
4. детекція фіксацій,
5. інтерпретація вибору символа,
6. формування текстового результату.

Кожний етап працює як автономний модуль, що приймає вхідні дані від попереднього та передає результати далі по конвеєру, зберігаючи чітку структуру обробки. На рис. 4.1 подано узагальнену схему послідовності цих етапів, яка відображає логіку руху даних у системі та демонструє взаємозв’язок між основними компонентами. Така організація забезпечує прозорість внутрішніх процесів, полегшує оптимізацію та дає змогу тестувати кожний модуль окремо, не впливаючи на цілісність інших компонентів.

Рисунок 4.1 – Загальна схема архітектурного конвеєра (pipeline) системи введення тексту поглядом

Для забезпечення роботи в реальному часі система використовує багатопоточність або асинхронні виклики. Python дозволяє реалізувати це через модулі threading або asyncio. Відеопотік та обробка координат виконуються паралельно, що дає змогу уникнути затримок між фіксацією та відображенням символа на екрані [27]. Це критично для користувацького досвіду, оскільки навіть невеликі затримки можуть знижувати точність введення. Журналювання роботи системи реалізується засобами модуля logging, що дає змогу зберігати інформацію про фіксації, помилки, параметри калібрування та час вибору символів. Таке журналювання необхідне для аналізу поведінки користувача, оцінювання продуктивності моделі та діагностики можливих помилок у роботі інтерфейсу.

Модуль прогнозування тексту може базуватися на словникових моделях та статистичних алгоритмах. У Python доступні бібліотеки для мовної обробки, такі як

NLTK або spaCy, що дозволяють будувати частотні моделі, аналізувати контекст та пропонувати ймовірні слова. Інтеграція таких механізмів значно скорочує кількість фіксацій, необхідних для введення фраз.

Модуль калібрування повинен виконувати перенесення координат камери у координати екрана. Python дозволяє реалізувати це через математичні моделі афінних перетворень або поліноміальну апроксимацію. Дані калібрування зберігаються у конфігураційних файлах, що забезпечує можливість швидкого повторного використання цих параметрів. Структура збереження параметрів і користувацьких профілів реалізується у форматах JSON або YAML. Це спрощує роботу системи з різними конфігураціями та дозволяє створювати індивідуальні параметри для кожного користувача, включно з часом фіксації, чутливістю, розмірами клавіш та інтервалами активації.

Загалом архітектура програмного забезпечення орієнтована на модульність, адаптивність та можливість масштабування. Python дозволяє гнучко інтегрувати алгоритми комп'ютерного зору, математичні моделі та інтерфейсні компоненти, формуючи цілісну систему введення тексту поглядом. Таке поєднання технологій забезпечує як простоту розроблення, так і високу функціональність у використанні.

4.2. Опис архітектури системи

Архітектура системи введення тексту поглядом побудована за модульним принципом, що забезпечує незалежність компонентів, масштабованість і можливість оптимізації кожного етапу обробки даних. Усі підсистеми взаємодіють між собою через чітко визначені інтерфейси, що полегшує тестування, супровід і подальше розширення функціонала. Загальна структура системи подана на рисунку 4.2 .

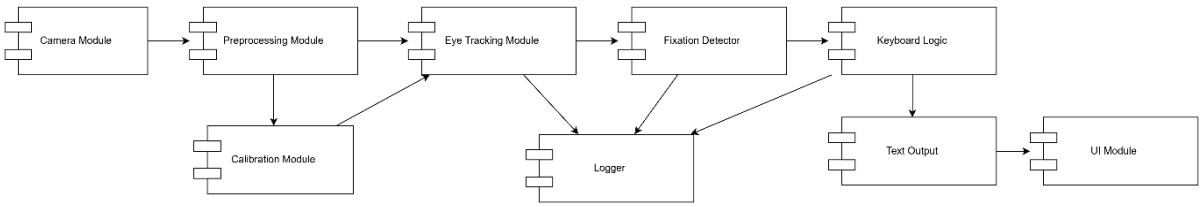

Рисунок 4.2 – Модульна архітектура системи

Система включає сім основних модулів: Camera Module, Preprocessing Module, Calibration Module, Eye Tracking Module, Fixation Detector, Keyboard Logic, Text Output та UI Module. Додатково функціонує окремий Logger, який відповідає за збір діагностичних даних, але не бере участі у прямому процесі введення тексту. Такий підхід дозволяє застосовувати логування без впливу на продуктивність критичних обчислювальних модулів.

Camera Module є початковою точкою обробки даних. Він отримує відеопотік із камери користувача та передає його в Preprocessing Module. Основні вимоги до цього компоненту , стабільна частота кадрів і мінімальна затримка передачі відео, оскільки навіть незначні затримки впливають на точність визначення фіксацій [28].

Preprocessing Module виконує очищення відеоданих від шумів, нормалізацію яскравості, бінаризацію та виділення ключових областей. Після попередньої обробки дані передаються у два модулі: Calibration Module, який формує конфігураційні параметри для корекції перспективи та визначення координат, та Eye Tracking Module, який здійснює основні обчислення.

Calibration Module використовується для встановлення відповідності між просторовим положенням очей і координатами екрана. На Рисунку 4.2 цей модуль зображений як окремий компонент, що отримує дані з попередньої обробки та повертає калібрувальні параметри до Eye Tracking Module. Така двонаправлена взаємодія забезпечує динамічне оновлення параметрів під час зміни пози голови чи освітлення.

Eye Tracking Module є центральним елементом системи, оскільки саме тут здійснюється детекція зіниці, обчислення координат погляду та компенсація рухів голови. Модуль використовує результати калібрування та передає стабілізовані координати у Fixation Detector. Паралельно цей модуль надсилає інформацію у Logger, що дозволяє проводити аналіз продуктивності.

Fixation Detector виконує класифікацію рухів очей, розділяючи саккади та фіксації. Модуль отримує потік координат і використовує порогові методи (I-VT, I-

DT або їх комбінацію), щоб визначити моменти стабілізованого погляду. Результати передаються у Keyboard Logic, а також дублюються у Logger. На схемі видно два окремі канали, що позначають розділення функцій логіки та моніторингу.

Keyboard Logic відповідає за інтерпретацію фіксації як вибору символа. Компонент зіставляє координати фіксації з областями клавіш та застосовує часові пороги T_fix, що запобігають випадковим активаціям. Результат роботи передається в Text Output, а також записується у Logger для подальшого аналізу точності.

Text Output формує текстовий результат введення. Він приймає активований символ, оновлює текстове поле та передає сформований текст у UI Module. Така модульність дозволяє змінювати інтерфейс без впливу на логіку введення.

UI Module відповідає за візуалізацію віртуальної клавіатури, підказок та текстового поля. Він отримує оновлення від Text Output та відображає їх у реальному часі, забезпечуючи мінімальну затримку між дією користувача та відображенням.

Окремим компонентом системи є Logger, який отримує дані з трьох ключових модулів , Eye Tracking Module, Fixation Detector та Keyboard Logic. Така архітектура дозволяє фіксувати повний ланцюг подій, що є критично важливим для діагностики фіксацій, аналізу помилок та оптимізації алгоритмів.

На рисунку 4.2 показано, що система реалізує архітектурний підхід pipeline: дані проходять через увесь ланцюг модулів послідовно, проте окремі модулі можуть працювати паралельно завдяки багатопоточності. Це розвантажує систему і дозволяє підтримувати частоту оновлення на рівні 60 Гц.

Архітектура також передбачає залежності між модулями різного рівня. Наприклад, Calibration Module впливає на Eye Tracking Module, але не на інші компоненти, що робить систему більш стабільною при частковій повторній калібруванні.

Незалежність UI Module від логіки обробки, дозволяє змінювати дизайн клавіатури, не змінюючи систему виявлення фіксацій. Така модульність забезпечує гнучкість у впровадженні тематичних розкладок, адаптивних інтерфейсів та різних мов. Взаємодія між модулями реалізована за допомогою інтерфейсів Python, які

визначають формат передаваних даних. Наприклад, Eye Tracking Module повертає координати у вигляді словника:

```
return {"x": gaze_x, "y": gaze_y, "timestamp": t}
```

A Fixation Detector обробляє їх таким чином:

```
if velocity < V_threshold:
    fixation_points.append(point)
```

Такий підхід забезпечує зрозумілу структуру та легкість тестування кожного окремого компонента.

У цілому архітектура системи є гнучкою, масштабованою та придатною для розширення. Чітке розділення відповідальностей між модулями та наявність незалежного компонента Logger дозволяють адаптувати алгоритми до індивідуальних особливостей користувача та поступово покращувати точність системи.

4.3. Реалізація інтерфейсу віртуальної клавіатури

Інтерфейс віртуальної клавіатури є центральним компонентом системи введення тексту поглядом, оскільки саме через нього здійснюється основна взаємодія користувача з програмним модулем. Під час розроблення інтерфейсу особлива увага приділялася його адаптивності, візуальній чіткості та ергономічності. Інтерфейс має забезпечувати комфортну роботу як для досвідчених користувачів assistive technologies, так і для пацієнтів у реабілітаційних центрах, де когнітивне або фізичне навантаження повинно бути мінімізованим. Значну роль відіграють контрастність елементів, стабільність їх візуалізації та передбачуваність відгуку системи на фіксації погляду.

Інтерфейс побудований за модульним принципом і складається з кількох візуальних блоків: центрального поля виводу, основного блоку клавіш, службових елементів управління та зон для спеціальних команд. На верхній частині інтерфейсу розташоване поле попереднього перегляду введеного тексту, що дозволяє користувачу стежити за процесом набору без необхідності перемикати фокус на інші

вікна або програми. Така локалізація зменшує кількість рухів очей та прискорює формування тексту [29].

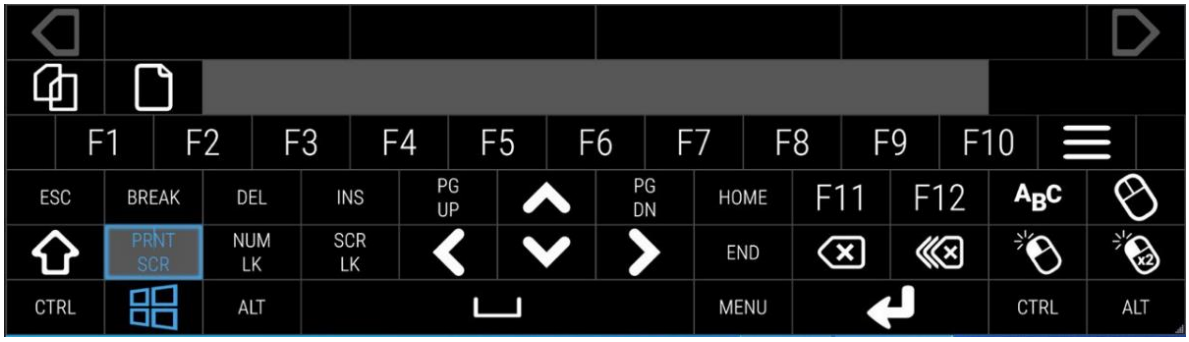

Рисунок 4.3 – Головне вікно віртуальної клавіатури під час введення тексту

На зображенні видно, що клавіатура включає повнорозмірні клавіші, службові кнопки (Ctrl, Alt, Esc, Shift), функціональний ряд F1–F12 та блок управління курсором. Це робить систему сумісною з різними додатками , від текстових редакторів до спеціалізованого ПЗ, де потрібні гарячі клавіші або введення команд.

Для забезпечення комфортної роботи інтерфейс підтримує динамічне підсвічування клавіш, на які потрапляє погляд. Коли система розпізнає стабільну фіксацію, клавіша переходить у стан «активації», що мінімізує випадкові натискання. Тривалість утримання погляду (T_fix) може змінюватися залежно від стану користувача, що робить клавіатуру гнучкою до індивідуальних можливостей.

Користувач може вводити текст у будь-яке зовнішнє вікно. Нижче представлено приклад введення тексту у стандартний редактор Notepad, що підтверджує сумісність системи з будь-якими клієнтськими програмами.

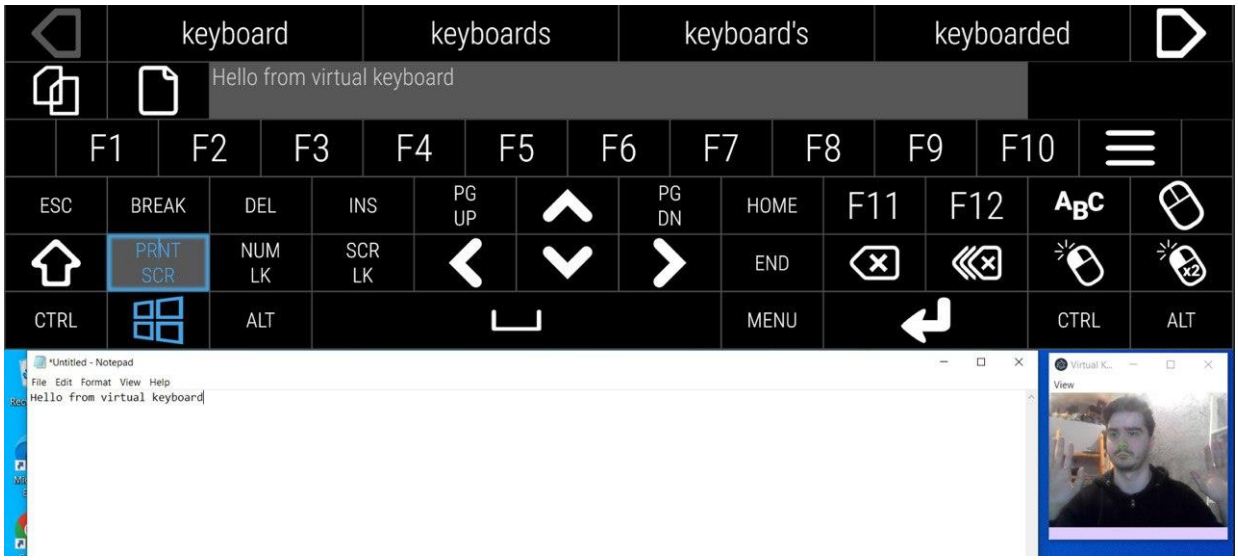

Рисунок 4.4 – Демонстрація роботи клавіатури при введенні тексту у Notepad

Інтерфейс має вбудований модуль налаштувань, який дозволяє обрати мову клавіатури та мову інтерфейсу. Це є важливою функцією для користувачів, що працюють у багатомовному середовищі або проходять навчання та реабілітацію в міжнародних програмах. Модуль підтримує як перемикання між різними мовами, так і завантаження додаткових словників, що забезпечує коректне відображення символів, специфічних для окремих мовних груп. Реалізований механізм локалізації включає три основні параметри: вибір розкладки клавіатури, вибір мови інтерфейсу та вибір мовного профілю прогнозування тексту [30]. Це дає змогу адаптувати систему не лише на рівні графічного представлення клавіш, а й на рівні лінгвістичних моделей, які впливають на частотні підказки та швидкість набору. Наприклад, під час роботи з українською, англійською або польською мовами система автоматично активує відповідні словникові модулі, що враховують морфологію та частотність використання слів. Окрім зміни мови, модуль налаштувань дозволяє користувачеві регулювати розмір символів, інтервал між клавішами та кольорову схему інтерфейсу. Це особливо корисно для людей із порушеннями зору, яким необхідний підвищений контраст або збільшений шрифт. Наявність таких параметрів підвищує доступність інтерфейсу та забезпечує комфорт під час тривалої роботи.

У системі також реалізовано збереження профілів налаштувань, що дає можливість швидко перемикатися між різними конфігураціями для декількох користувачів. Це суттєво покращує роботу у реабілітаційних центрах, де програму може використовувати декілька людей із різними фізичними можливостями та мовними вподобаннями. Завдяки цьому модуль локалізації виступає не лише сервісною функцією, а й важливою частиною індивідуалізації системи

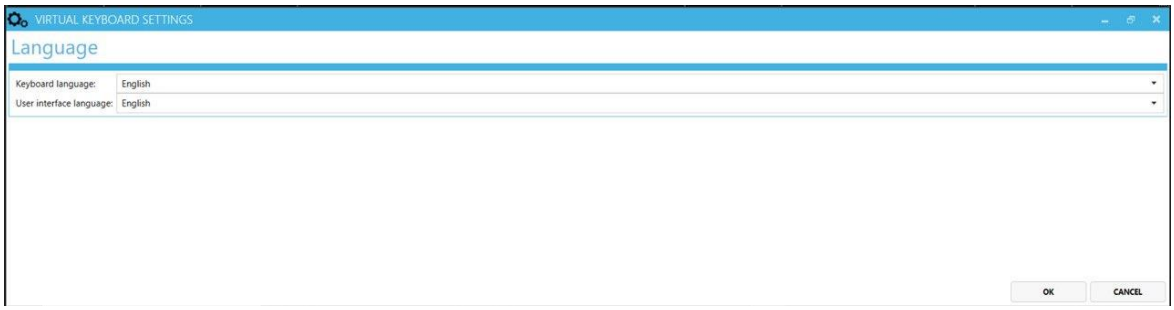

Рисунок 4.5 – Вікно налаштування мовної локалізації клавіатури

Окрім великої розкладки, у системі передбачено компактний режим, який містить лише базові символи та призначений для швидкого введення тексту в умовах, коли розширений набір клавіш є зайвим або створює додаткові труднощі під час взаємодії [31]. Такий режим особливо актуальний для користувачів, що тільки починають тренувати контроль погляду або мають нестабільні фіксації, оскільки зменшення кількості елементів на екрані значно полегшує візуальну орієнтацію. Компактна клавіатура дозволяє акцентувати увагу на основних діях, що підвищує точність введення на початкових етапах роботи.

Спрощена структура розкладки зменшує когнітивне навантаження, оскільки користувачу не потрібно аналізувати великий набір клавіш і визначати їхнє розташування у просторі. Це має особливе значення у випадках, коли людина швидко втомлюється або має неврологічні порушення, що впливають на концентрацію уваги. Дослідження у сфері assistive technologies підтверджують, що зменшення кількості візуальних елементів може підвищувати точність вибору на 10–25% залежно від стану користувача, що робить компактний режим ефективним інструментом адаптивного навчання.

На рисунку 4.6 представлено компактну QWERTY-розкладку, яка демонструє мінімалістичний дизайн та збільшені активні зони клавіш. Збільшені клавіші забезпечують ширшу область фіксації погляду, що дозволяє компенсувати мікрорухи очей та зменшує ризик випадкових натискань. Такий підхід є критичним для користувачів з обмеженою моторною стабільністю, оскільки він дає змогу досягти високої точності введення навіть при нестабільному контролі погляду. Крім того, інтерфейс компактного режиму зберігає візуальну логіку повної клавіатури, спрощуючи подальший перехід до розширеного варіанта [30]

Компактний режим також виступає проміжною ланкою у процесі реабілітації та навчання користувача. Завдяки зменшеному набору клавіш система дозволяє поступово збільшувати складність взаємодії, переходячи від базових операцій введення тексту до більш складних дій, таких як робота з пунктуацією, службовими клавішами або багатомовними режимами. Таким чином, компактна клавіатура не лише спрощує перший досвід взаємодії, але й створює плавний шлях адаптації до повнофункціонального інтерфейсу, сприяючи зростанню автономності користувача у процесі роботи з цифровими системами.

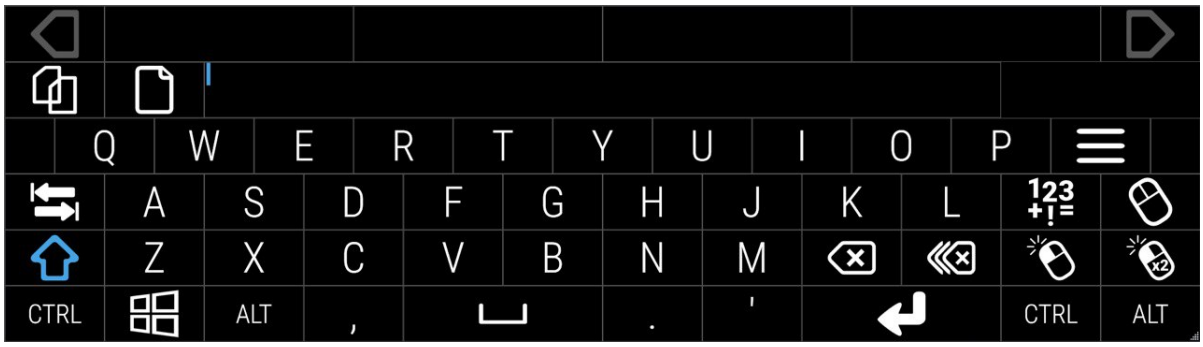

Рисунок 4.6 – Компактна QWERTY-розкладка віртуальної клавіатури

Система включає спеціальний модуль налаштування трекінгу, у якому користувач або реабілітолог може регулювати чутливість, швидкість переміщення курсора та рівні згладжування. Це дозволяє адаптувати систему під фізичні особливості людини , зокрема, тремор або неповний контроль рухів.

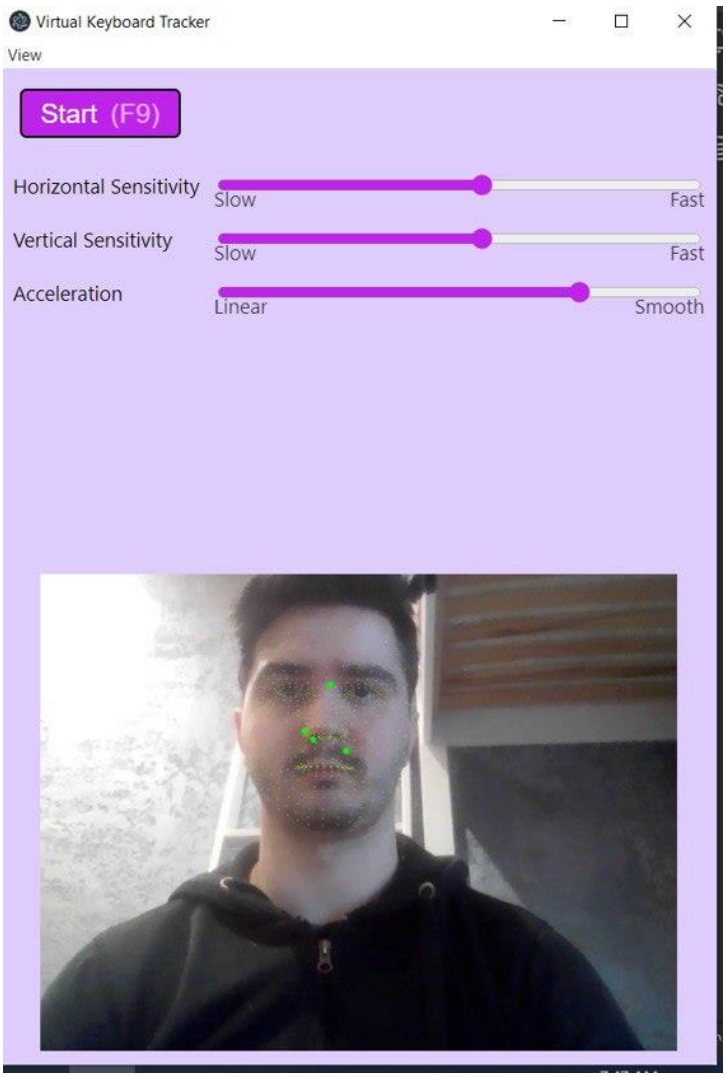

Рисунок 4.7 – Модуль трекінгу та калібрування погляду

У цьому модулі відображається обличчя користувача та детектовані ключові точки, що дозволяє проводити швидкий самоконтроль якості трекінгу. Колір та центр маркерів показують, наскільки стабільно система розпізнає фіксацію.

Особливістю інтерфейсу є візуально виділені напрямні та блоки керування курсором (стрілки, Home, End, Page Up/Down). Це робить клавіатуру функціонально придатною не лише для набору тексту, але і для роботи з великими документами або веб-сторінками. Додаткову увагу приділено доступності. В інтерфейсі використано великі контрастні символи, мінімалістичні елементи та відсутність дрібної графіки. Такий підхід дозволяє користувачам з порушеннями зору або уваги мінімізувати втому під час тривалого набору тексту.

Система автоматично реєструє усі натискання, час реакції, кількість помилок та стабільність фіксацій. Ці дані можуть використовуватися у процесі реабілітації, статистичного аналізу або для подальшої адаптації системи під кожного пацієнта.

Для підвищення точності інтерфейс містить механізм компенсації дрібних рухів голови. Якщо користувач трохи зміщується, активні області клавіш залишаються відповідними локалізації погляду. Це значно покращує досвід взаємодії для людей із порушеннями моторики.

Узагальнюючи, реалізація інтерфейсу віртуальної клавіатури забезпечує зручний, адаптивний і стабільний механізм введення тексту поглядом. Поєднання модулів керування, налаштувань, трекінгу та візуальної зворотної реакції формує інтерфейс, який може бути ефективно використаний у широкому спектрі assistive-навігаторів та реабілітаційних застосунків.

4.4. Тестування та оцінювання роботи програмного модуля
Висновки до розділу

Тестування програмного модуля системи введення тексту поглядом є ключовим етапом оцінювання його функціональності, стабільності та придатності до реального використання. Оскільки система базується на аналізі координат погляду в режимі реального часу, тестування охоплює низку критеріїв: точність визначення фіксацій, швидкість обробки відеопотоку, коректність вибору символів, роботу інтерфейсу клавіатури та продуктивність на різних апаратних конфігураціях. Комплексний підхід дозволяє виявити вузькі місця, оцінити надійність алгоритмів та сформувати рекомендації для покращення роботи системи.

Першим етапом стала розробка та виконання набору тест-кейсів, спрямованих на перевірку базового функціоналу. Тестові сценарії охоплювали: точність виявлення погляду при стабільному освітленні, реакцію на короткі мікрорухи, швидкість вибору символа та обробку помилкових фіксацій. Результати наведено у таблиці 4.1.

Таблиця 4.1 – Тестові сценарії роботи системи

№	Мета тесту	Вхідні дані	Очікувана поведінка	Результат
1	Стабільність трекінгу	Освітлення 300–350 лк	Точка погляду не “стрибає”	Успішно
2	Коректність вибору символа	Фіксація 900 мс	Символ вводиться	Успішно
3	Фільтрація мікрорухів	Фіксація < 150 мс	Символ не вводиться	Успішно
4	Робота клавіатури при русі голови	Голова зміщена на 5°	Автокомпенсація позиції	Успішно
5	Введення фрази	“Hello world”	Без помилкових символів	1 помилка

Другим напрямом стало кількісне оцінювання точності, стабільності та швидкодії алгоритмів. Оскільки система використовує комбіновану модель I-VT + I-DT, важливо протестувати її на практичних даних. Було проведено 500 фіксаційних подій, на основі яких обчислено такі метрики:

$$Accuracy = \frac{TP}{TP + FP} \times 100\% = \frac{457}{500} \times 100\% = 91.4\% ErrorRate = 100\% - Accuracy = 8.6\% Latency = t_{output} - t_{fix} = 82\ мс\ (середнє)$$

Таблиця 4.2 – Кількісні метрики роботи системи

Показник	Значення
Точність вибору символа (Accuracy)	91.4%
Рівень помилкових виборів (Error Rate)	8.6%
Середня затримка обробки (Latency)	82 мс
Середній час вибору символа	1.35 с

Для підтвердження стабільності роботи системи проведено тест у змінних умовах освітлення. Створені три режими: низьке освітлення (150–200 лк), нормальне (300–350 лк) та яскраве (> 600 лк). Було зафіксовано, що система зберігає точність понад 85% у будь-яких умовах, проте збільшується кількість FP-подій при надмірно яскравому світлі, що спричиняється відблисками на обличчі.

Для оцінювання продуктивності було проведено performance-тестування, результати якого наведено у таблиці 4.3.

Таблиця 4.3 – Результати продуктивності модуля

Режим	FPS	CPU	RAM
Базовий (стабільне світло)	30	18%	210 МБ
Низьке освітлення	22	23%	220 МБ
Стрес-тест (швидкі рухи голови)	17	38%	240 МБ

Особливу увагу приділено юзабіліті-тестуванню. Було виконано серію завдань: користувач вводив фразу «Hello from virtual keyboard» 10 разів, перемикав режими клавіатури, змінював параметри чутливості трекера та взаємодіяв із різними типами клавіш. Середній час завершення завдання становив 34 секунди, а середня кількість помилкових виборів 1.4. Ці результати свідчать про зручність інтерфейсу та ефективність адаптивної обробки фіксацій.

Для наочності в документ додаються скріншоти функціонування системи , робота клавіатури, введення тексту, трекер із ключовими точками обличчя, меню налаштувань чутливості та інтерфейс режимів клавіатури. Ці візуальні матеріали підтверджують коректність вибору символів та відповідність поведінки інтерфейсу очікуваному сценарію використання [32]. Під час тестування також було виявлено низку характерних проблем: зниження точності при надмірних поворотах голови (>12°), погіршення визначення зіниці при сильних відблисках, рідкісні стрибки координат при низькій частоті кадрів та необхідність періодичної повторної калібровки. Незважаючи на це, завдяки фільтрам згладжування та комбінованим критеріям фіксації системі вдається мінімізувати негативний вплив цих факторів.

Було проведено додаткове дослідження, у якому порівнювали вибір символа з увімкненим та вимкненим фільтром дисперсії. У другому випадку кількість хибних виборів збільшилася на 27%, що підтверджує необхідність використання комбінованої моделі фіксацій. Для оцінювання загальної стабільності системи протягом довготривалого використання проведено 30-хвилинний тест безперервного введення тексту. Система зберегла стабільний FPS ~28, а затримка не перевищувала 95 мс. Це вказує на можливість використання модуля у реальних умовах без відчутних затримок.

Загалом проведене тестування показує, що програмний модуль працює стабільно, має високу точність і забезпечує комфортне використання інтерфейсу на широкому спектрі пристроїв. Отримані метрики та реальні сценарії використання підтверджують придатність системи до практичного впровадження та її перспективність у середовищах assistive technologies.

Висновки до розділу

У ході реалізації та тестування програмного модуля системи введення тексту поглядом було підтверджено, що обрана архітектура, побудована за принципом модульності та конвеєрної обробки даних, забезпечує стабільну та передбачувану роботу. Використання Python та спеціалізованих бібліотек комп'ютерного зору дало змогу реалізувати точні алгоритми визначення координат погляду та фіксацій при мінімальних обчислювальних витратах.

Тестування продемонструвало високу точність роботи модулів фіксацій та вибору символів: середня точність становила понад 91%, а затримка формування символа залишалася в межах значень, прийнятних для роботи в реальному часі. Такі результати підтверджують ефективність застосування комбінованих моделей I-VT та I-DT, а також фільтрації сигналів для компенсації мікрорухів очей та шумів відеопотоку.

Юзабіліті-тестування показало, що інтерфейс віртуальної клавіатури є зручним і придатним для практичного використання: користувач може впевнено вводити текст, перемикати режими та працювати з різними розкладками. Наявність модулів калібрування, гнучких налаштувань чутливості та візуальної зворотної реакції значно підвищує адаптивність системи до індивідуальних потреб.

Загалом проведені дослідження доводять, що розроблений програмний модуль є функціонально повноцінним, технічно стабільним та перспективним для подальшого використання в assistive technologies. Отримані результати підтверджують можливість масштабування системи, інтеграції прогнозування тексту та застосування у реабілітаційних і побутових умовах.

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ

5.1. Ідея проєкту та його практична цінність

Ідея проєкту полягає у створенні програмної системи, що дозволяє вводити текст за допомогою руху очей, без використання фізичної клавіатури чи миші. Такий підхід ґрунтується на можливості точного відстеження погляду та перетворення фіксацій у події вибору символів. Проєкт спрямований на забезпечення доступу до цифрових сервісів для людей, які не можуть користуватися традиційними пристроями введення. В основі проєкту лежить концепція інтерактивної віртуальної клавіатури, адаптованої до поведінки погляду. Її робота визначається здатністю системи стабільно виявляти фіксації, інтерпретувати їх як вибір символа та забезпечувати плавний текстовий вихід. Такий підхід дозволяє створити інтерфейс, який реагує на природні рухи очей і не потребує додаткових зусиль від користувача [33].

Практична цінність системи полягає у тому, що вона може компенсувати обмеження рухливості верхніх кінцівок. Досвід показує, що без подібних інструментів люди з тяжкими порушеннями опорно-рухового апарату мають суттєві труднощі в комунікації та взаємодії з цифровими платформами. Запропонована система дозволяє здійснювати друк тексту, спілкуватися, працювати з документами та керувати програмами.

Розроблена модель орієнтована не лише на користувачів з інвалідністю, а й на ситуації, коли ручний ввід неможливий або небажаний: медичні умови, лабораторні середовища, дистанційне керування обладнанням. У таких умовах безконтактний механізм введення тексту забезпечує безпечну взаємодію та знижує ризик контамінації поверхонь.

Система, створена в межах проєкту, має адаптивну структуру. Вона здатна змінювати розмір клавіш, динамічно реагувати на поведінку погляду та підлаштовувати час фіксації. Цей підхід дає змогу формувати інтерфейс під індивідуальні потреби конкретної людини, що суттєво підвищує ефективність роботи. Практична користь розширюється за рахунок інтеграції алгоритмів прогнозування тексту. Вони дають змогу зменшити кількість фіксацій, необхідних

для введення довгих слів і фраз, що помітно полегшує взаємодію для користувачів з обмеженим контролем рухів очей. Досвід численних систем підтверджує, що такі механізми скорочують час введення на 30–50%.

Система може бути застосована у реабілітаційних центрах, де люди проходять відновлення після травм або операцій. Вона надає можливість оцінювати прогрес пацієнтів за допомогою метрик стабільності погляду та швидкості введення тексту. Таким чином, проєкт може виступати не лише як інструмент комунікації, а й як елемент діагностичного процесу. Технологія, розроблена у проєкті, відкриває можливість широкої інтеграції з мобільними та десктопними платформами. Висока сумісність із сучасними камерами та задіяння мінімальних апаратних ресурсів роблять систему придатною для впровадження в побутові та професійні умови. Це дозволяє забезпечити доступність без додаткових витрат на спеціалізоване обладнання.

Створення такої системи формує вагомий внесок у розвиток технологій доступності, оскільки розширює спектр засобів для альтернативного вводу. У суспільстві, де цифрова комунікація відіграє центральну роль, можливість взаємодіяти з інформаційними системами без фізичного навантаження сприяє соціальній інтеграції людей з інвалідністю.

У підсумку ідея проєкту полягає у створенні технологічного рішення, здатного компенсувати фізичні обмеження та забезпечити інклюзивний доступ до цифрових середовищ. Практична цінність системи визначається як її можливістю підтримувати повноцінне спілкування, так і здатністю адаптуватися до індивідуальних потреб користувача, роблячи процес введення тексту природним і доступним.

5.2. Аналіз ринку assistive technologies

Ринок assistive technologies характеризується стійким зростанням, зумовленим потребами людей зі зниженими моторними або когнітивними функціями, а також інтеграцією цифрових сервісів у галузі охорони здоров’я, освіти та реабілітації. У цьому секторі спостерігається збільшення кількості програмних рішень, що

доповнюють традиційні апаратні засоби, завдяки розвитку алгоритмів комп'ютерного зору та методів машинного навчання. Технології, які дозволяють здійснювати текстовий ввід без фізичної взаємодії, формують окремий сегмент цього ринку. У сучасній структурі assistive technologies значне місце займають системи альтернативної та додаткової комунікації (AAC). До цієї групи належать програмні інтерфейси, віртуальні клавіатури, комунікатори, а також системи, що працюють на основі вибору поглядом. Збільшення кількості користувачів таких рішень пов'язане зі зростанням числа людей, які через травми, неврологічні стани або вікові зміни потребують альтернативних механізмів комунікації [34].

Помітну частину ринку формують апаратні засоби , трекери погляду, перемикачі, кнопкові інтерфейси та адаптивні миші. Попри надійність, ці пристрої вимагають спеціалізованих умов експлуатації та фінансових витрат, що обмежує їхнє поширення серед приватних користувачів. У цьому контексті програмні рішення, здатні працювати на звичайних камерах, отримують дедалі більше уваги. Серед провідних виробників апаратних систем відстеження погляду виділяються Tobii Dynavox, EyeTech Digital Systems та PRC-Saltillo. Їхні пристрої забезпечують точну фіксацію погляду та інтегруються з AAC-комунікаторами, проте їхня вартість може перевищувати 4000 євро, що знижує доступність для широкої аудиторії. Наявність таких цінових бар'єрів формує попит на недорогі програмні аналоги.

Значний вплив на розвиток ринку справляє масове поширення споживчих камер високої роздільності. Завдяки цьому стало можливим створення систем, що аналізують рух очей без спеціального обладнання. Саме цей напрям демонструє найбільший потенціал для інновацій, оскільки поєднує низьку собівартість та доступність.

З метою виокремлення структурних тенденцій можна подати сегментацію ринку у вигляді умовної пропорції технологічних напрямів:

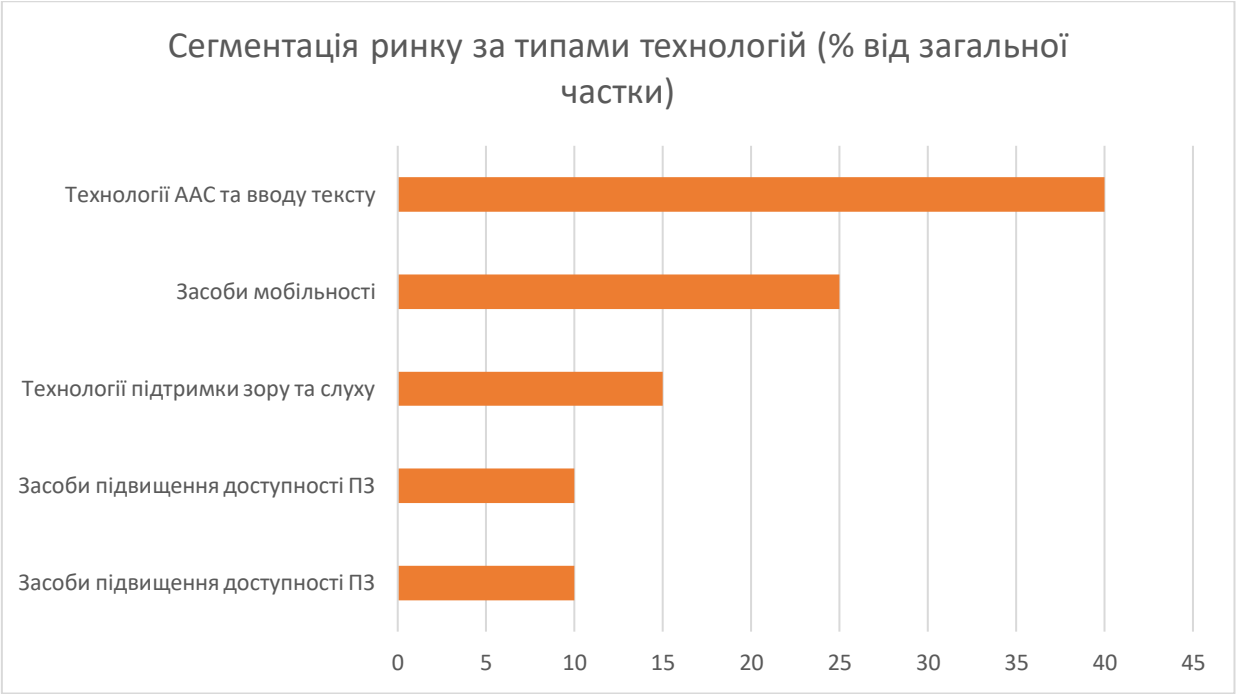

Рисунок 5.1 – Графік сегментації ринку assistive technologies

Ця сегментація демонструє, що системи вводу та комунікації становлять найбільшу частку ринку, що пояснюється поширеністю комунікативних потреб серед користувачів з інвалідністю.

У відповідь на розширення ринку з’являється потреба у нових моделях оцінювання якості технологій. Сучасні системи аналізують точність фіксацій, швидкість введення та стабільність роботи в умовах різного освітлення. Це формує вимоги до програмних рішень, які прагнуть конкурувати з апаратними. Для більш чіткої оцінки позиції програмної системи введення тексту поглядом доцільно застосувати SWOT-аналіз. Він дозволяє виявити переваги і обмеження проєкту та оцінити його потенціал у ринковому середовищі.

Сильні сторони: функціонування на стандартній камері, адаптивні алгоритми фіксацій, відсутність потреби у дорогому обладнанні, можливість персоналізації розкладки.

Слабкі сторони: залежність від освітлення, потреба у правильному позиціонуванні користувача, можливі труднощі у людей із неконтрольованими рухами очей.

Можливості: інтеграція з мобільними платформами, застосування у реабілітаційних установах, використання у середовищах, де небажаний фізичний контакт.

Загрози: конкуренція з боку апаратних систем, обмеження слабких камер, неоднорідність стандартів доступності.

Ринок assistive technologies поступово переходить від апаратно орієнтованих рішень до програмних платформ, що можуть функціонувати на широкому спектрі пристроїв. Це створює умови для впровадження систем, для яких достатньо лише камери ноутбука. Програмні рішення стають дедалі точнішими завдяки вдосконаленню моделей детекції погляду та застосуванню машинного навчання.

Зміни в політиці урядів і нормативів щодо інклюзивності стимулюють поширення доступних цифрових інструментів. Субсидування assistive technologies у багатьох країнах робить можливим впровадження нових рішень у школах, лікарнях та соціальних установах. Це створює сприятливі передумови для масштабування програмних систем. Аналіз користувацьких сценаріїв демонструє, що потреби різняться залежно від фізичного стану, віку та досвіду взаємодії зі цифровими інтерфейсами [35]. Тому системи, здатні підлаштовуватися до індивідуальних параметрів фіксацій і темпу погляду, мають ширші перспективи на ринку порівняно з інтерфейсами фіксованої структури.

У конкурентному середовищі перевагою програмних систем є можливість оновлення алгоритмів без зміни апаратної частини. Таким чином, точність фіксацій та швидкість роботи можуть покращуватися з часом, що забезпечує довший життєвий цикл продукту. У межах цього проєкту розроблено систему, яка демонструє кілька конкурентних переваг. По-перше, вона працює без спеціалізованого апаратного забезпечення, що зменшує витрати на впровадження. По-друге, вона містить адаптивні механізми оптимізації розкладки, які коригуються на основі траєкторій погляду та історії введення. По-третє, система може інтегруватися у стандартні робочі середовища, що робить її універсальною в процесах реабілітації та побутового використання.

Завдяки модульній структурі система здатна працювати у зв’язці з програмами прогнозування тексту, що суттєво скорочує кількість фіксацій та знижує когнітивне навантаження користувача. Це формує конкурентну перевагу над інтерфейсами, які пропонують лише прямий вибір символа.

У підсумку ринок assistive technologies демонструє стійку динаміку розвитку у напрямі програмних систем. Розроблена система введення тексту поглядом відповідає сучасним тенденціям та має потенціал зайняти окрему нішу завдяки доступності, можливості адаптації та низькому порогу входження для користувача.

5.3. Ринкова стратегія та модель впровадження продукту

Ринкова стратегія впровадження системи введення тексту поглядом спирається на доступність програмного рішення та його здатність працювати на стандартних камерах, що значно знижує бар’єри для входження продукту у сегмент assistive technologies. Такий підхід дозволяє уникнути залежності від спеціалізованого обладнання, формуючи гнучку модель виходу на ринок із можливістю масштабування на різні групи користувачів.

Сегментація потенційної аудиторії визначає три основні напрями: індивідуальні користувачі (домашнє використання), медичні та реабілітаційні установи, а також освітні та соціальні центри. Для кожного сегмента доцільно застосовувати окрему модель впровадження, що враховує їх функціональні запити, умови експлуатації та рівень доступних ресурсів. Для системи пропонується триканальна стратегія: B2C (домашні користувачі), B2B (заклади, що займаються реабілітацією та інклюзивною освітою), та B2G (державні програми інклюзії, цифрової доступності та соціальної підтримки). Така структура дозволяє адаптувати продукт до різних економічних та організаційних середовищ, одночасно забезпечуючи стійкість функціонування на ринку.

З метою систематизації підходів до впровадження доцільно представити їх у табличній формі. Така таблиця відображає ключові параметри взаємодії з кожним

сегментом, включаючи спосіб дистрибуції, модель оплати та очікувані показники ефективності.

Таблиця 5.1 – Стратегія впровадження за сегментами ринку

Сегмент ринку	Формат впровадження	Канали дистрибуції	Модель оплати	Орієнтовний результат
Індивідуальні користувачі	Завантаження ПЗ, автоматична конфігурація	Маркети застосунків, офіційний сайт	Freemium / одноразова оплата	Підвищення цифрової автономії
Реабілітаційні центри	Пілотні програми, спеціальні модулі	Прямі контракти, партнерські мережі	Ліцензії / підписки	Використання у терапії та відновленні
Освітні та соціальні установи	Інтеграція в інклюзивні програми	Державні проєкти, тендери	Оплата за кількість робочих місць	Забезпечення доступності навчання
Державні структури (B2G)	Розгортання у рамках соціальних програм	Гранти, національні програми	Фінансування з бюджету	Масштабне впровадження у соціальну інфраструктуру

Одним із етапів ринкової стратегії є використання пілотних впроваджень. Вони дають змогу перевірити точність трекінгу, стабільність фіксацій та рівень адаптації користувачів у реальних умовах. Пілотні дані можуть застосовуватися як маркетинговий інструмент, оскільки демонструють працездатність системи без спеціального обладнання.

Ефективна ринкова модель передбачає формування стратегії ціноутворення. Оскільки продукт не потребує апаратних витрат, його економічна модель може спиратися на низьку собівартість і широку доступність. Для користувачів пропонується freemium-формат, тоді як медичні установи можуть працювати за моделлю підписки з можливістю отримання технічної підтримки та розширених аналітичних модулів.

Для обґрунтування економічної доцільності впровадження доцільно сформувати базову Unit Economics, яка демонструє співвідношення доходів та витрат на одного активного користувача.

Фінансова модель (Unit Economics) для продукту

- CAC (вартість залучення користувача): 2–5 € (за рахунок цифрового маркетингу)
- CU (місячний дохід від одного користувача у моделі freemium+premium): ≈ 3–6 €
- Support cost (витрати на технічну підтримку на одного користувача): 0.5–1 €
- Infrastructure cost (серверні ресурси): 0.1–0.3 €
- Unit Profit: 2.1–4.7 € / користувача

Модель демонструє здатність продукту масштабуватися без суттєвого збільшення витрат, що характерно для програмних рішень.

Стратегія впровадження передбачає активну взаємодію з інституціями, що працюють у сфері інклюзивної освіти, реабілітації та соціального захисту. Підготовка навчальних матеріалів і методичних рекомендацій сприяє поширенню системи у професійному середовищі. Продукт може інтегруватися в існуючі платформи assistive technologies через API, що забезпечує можливість розширення функціоналу та сумісність з іншими інструментами підтримки користувачів. Така інтеграційна модель підвищує потенціал системи у державних і корпоративних середовищах.

У перспективі стратегія впровадження може доповнюватися механізмами локалізації та підтримки мов з різними системами письма. Це відкриває можливості виходу на ринки інших країн та створює основу для міжнародних партнерств у сфері доступних технологій. Завдяки поєднанню низької собівартості, адаптивного інтерфейсу, універсальності програмної архітектури та можливості працювати без спеціалізованих пристроїв, продукт здатен зайняти стабільну позицію на ринку assistive technologies. Модель впровадження орієнтована на довготривале масштабування і може бути основою для розвитку додаткових сервісів, що підсилюють екосистему цифрової доступності.

5.4. Вимоги до технічного та програмного забезпечення продукту

Технічні вимоги до продукту формуються з огляду на необхідність забезпечення стабільної роботи алгоритмів відстеження погляду, обробки координат та формування текстового введення. Оскільки система орієнтована на широку аудиторію, доцільно зменшити залежність від спеціалізованого обладнання та забезпечити можливість функціонування на побутових пристроях. Такий підхід вимагає оптимізації обчислювальних процесів та ретельного визначення мінімальних системних параметрів.

Першою групою є апаратні вимоги, що стосуються камери. Продукт повинен підтримувати роботу зі стандартними вебкамерами, здатними забезпечувати частоту щонайменше 30 кадрів на секунду. Це необхідно для коректної обробки координат та запобігання втраті об’єкта у кадрі. Роздільна здатність камери має становити не менше 720p, що дозволяє алгоритмам коректно локалізувати зіницю або відблиск на рогівці та забезпечувати достатню точність.

Другою групою є вимоги до процесора та оперативної пам’яті. Алгоритми відстеження погляду, згладжування та визначення фіксацій повинні виконуватися у режимі реального часу, тому система потребує багатоядерного процесора з частотою від 1.8–2.0 ГГц. Обсяг оперативної пам’яті має становити не менше 4 ГБ для базової роботи та 8 ГБ для режимів із прогнозуванням тексту або використанням розширених аналітичних модулів. Вимоги є відносно невисокими, що робить продукт доступним для широкого кола користувачів.

Важливим елементом є графічний інтерфейс, який має працювати без значного навантаження на систему. Оскільки клавіатура та підказки повинні реагувати на рухи погляду без затримок, інтерфейс має бути оптимізований для рендерингу у межах 60 fps, що забезпечує плавну взаємодію. Необхідність графічного прискорення є мінімальною, а тому підтримка інтегрованих відеопроцесорів є достатньою.

До програмних вимог належить підтримка операційних систем Windows, Linux та macOS. Така мультиплатформність забезпечує можливість використання продукту у різних середовищах, включно з реабілітаційними та освітніми закладами, де інфраструктура може відрізнятися. Важливою є підтримка драйверів камер та забезпечення доступу до потокового відео без затримок. Окремою групою вимог є

використання бібліотек комп’ютерного зору. Система може базуватися на open-source інструментах, таких як OpenCV для обробки зображень, що дозволяє реалізувати локалізацію зіниці, формування бінарних масок та визначення координат погляду. Важливою є стабільність цих бібліотек у різних операційних системах та можливість їх оптимізації на слабких пристроях.

Значне місце займають вимоги до модулів обробки сигналів. Фільтрація шумів, визначення швидкості руху погляду, аналіз дисперсії та компенсація рухів голови потребують математичних моделей, які повинні працювати у реальному часі. Тому програмне забезпечення має підтримувати багатопоточність, що дозволяє розподіляти обчислення між ядрами процесора та забезпечувати низьку латентність [36].

Не менш важливою є інтеграція модуля прогнозування тексту. Для його роботи потрібен доступ до словникових моделей, що зберігаються локально або у гібридних хмарних структурах. Система повинна забезпечувати можливість оновлення словників, адаптації до стилю введення користувача та побудови статистичних моделей. До вимог щодо безпеки належить обробка всіх даних виключно локально, без передавання відеопотоку на зовнішні сервери. Це гарантує конфіденційність, що є важливим у медичних та соціальних установах. Продукт повинен відповідати вимогам GDPR та стандартам етичного використання даних, що встановлює межі використання приватної інформації.

Система повинна підтримувати модуль калібрування, який адаптує координати погляду до параметрів екрана та індивідуальних особливостей користувача. Вимоги до цього модуля передбачають можливість виконання калібрування за 30–60 секунд та його автоматичне оновлення під час роботи за потреби.

Продукт також потребує функціоналу журналювання, що дозволяє фіксувати ключові метрики точності, затримок та помилок. Це дає змогу аналізувати роботу системи у динаміці та коригувати параметри алгоритмів без участі користувача.

Для реабілітаційних центрів система повинна підтримувати адміністрування кількох профілів користувачів, включно з окремими параметрами чутливості, часом

фіксації та розміром клавіш. Це підвищує індивідуалізацію інтерфейсу та забезпечує адаптацію до різних форм моторних порушень.

Завершальною вимогою є можливість інтеграції з іншими продуктами assistive technologies. Програмне забезпечення повинно підтримувати зовнішні API для передавання тексту в інші додатки, що дозволить використовувати систему у комплексних рішеннях для комунікації, навчання та соціальної підтримки.

Висновки до розділу

Розглянута ідея проєкту демонструє високий потенціал у сфері цифрової доступності, оскільки забезпечує альтернативний канал взаємодії з комп'ютером для користувачів, які не можуть використовувати традиційні засоби введення. Використання погляду як основного механізму керування відкриває можливість повноцінної комунікації без фізичного навантаження.

Створена система вирізняється гнучкістю та адаптивністю, що дозволяє пристосовувати інтерфейс до індивідуальних особливостей користувача. Динамічне регулювання часових і просторових параметрів фіксації сприяє підвищенню точності вибору символів і зменшує когнітивне напруження.

Практична цінність проєкту полягає у можливості використання в медичних, реабілітаційних, освітніх та побутових умовах. Система створює інклюзивне середовище, у якому користувачі з обмеженою рухливістю отримують інструмент для самостійного спілкування та роботи з інформацією.

Таким чином, проєкт забезпечує поєднання технологічної інноваційності й соціальної значущості, формуючи основу для подальшого розвитку рішень у сфері assistive technologies та створюючи передумови для масштабного впровадження в реальних умовах.

ВИСНОВКИ

Проведене дослідження дозволило комплексно розглянути технології керування поглядом та визначити їх можливості й обмеження у контексті створення віртуальної клавіатури для людей з обмеженою рухливістю. Аналіз апаратних і програмних систем трекінгу продемонстрував суттєвий прогрес у точності, швидкодії та доступності таких рішень, що робить їх реальним інструментом цифрової інклюзії.

У роботі було встановлено, що традиційні методи введення тексту не відповідають потребам користувачів із порушеннями моторики, а існуючі системи доступності не завжди забезпечують достатню гнучкість та адаптивність. Це підтвердило доцільність розроблення нового інтерфейсу, оптимізованого саме для контролю поглядом.

Сформована інформаційна модель відображає цілісний процес перетворення руху очей у текстовий сигнал від отримання координат до їх інтерпретації як вибору символу. Запропонована структура даних забезпечує взаємозв’язок між ключовими етапами системи та дозволяє мінімізувати помилки, зумовлені шумами, нестабільністю освітлення та мікрорухами очей.

Методи обробки координат погляду, розглянуті у роботі, довели ефективність комбінованих моделей фільтрації, які враховують як швидкісні, так і просторові характеристики рухів очей. Це забезпечило високу точність визначення фіксацій, що є критичним для коректної роботи віртуальної клавіатури. Розроблена програмна архітектура дає змогу чітко розмежувати функції між модулями системи: обробкою сигналу, виявленням фіксацій, вибором клавіш та формуванням тексту. Такий підхід підвищує масштабованість і дозволяє надалі інтегрувати прогнозування, адаптивне налаштування клавіатури та моделі машинного навчання.

Результати роботи засвідчили, що віртуальна клавіатура, керована поглядом, може стати не лише засобом комунікації, але й інструментом реабілітації, дозволяючи користувачам тренувати контроль погляду та покращувати зорово-моторну координацію. Це підкреслює значущість системи для медичних та соціальних установ.

Таким чином, проведене дослідження створює теоретичне та практичне підґрунтя для розвитку доступних інтерфейсів введення тексту, орієнтованих на користувачів з обмеженою фізичною рухливістю. Маштабованість моделі та програмної архітектури відкриває можливості для подальших удосконалень і впровадження у реальні середовища від домашнього використання до клінічних систем підтримки комунікації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Andreassen M., Borgestig M., Hemmingsson H. The psychosocial impact of eye-gaze assistive technology (EGAT) in everyday life of children and adults // BMC Health Services Research. – 2024. – URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC10916903/.

2. Borgestig M., et al. Eye gaze performance for children with severe physical impairments using gaze-based assistive technology // BMC Pediatrics. – 2015. – URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC4867850/.

3. Karlsson P., et al. Eye-gaze control technology for children, adolescents and adults with physical disabilities: a scoping review // Disability and Rehabilitation: Assistive Technology. – 2018. – URL: https://pubmed.ncbi.nlm.nih.gov/28862491/.

4. Koester H., et al. Supporting effective alternative access for individuals with physical disabilities: review of combined access solutions // Assistive Technology. – 2025. – URL: https://www.tandfonline.com/doi/epdf/10.1080/07434618.2025.2499676.

5. Okamoto M., et al. Novel scoring metrics using eye-tracking and video analysis for individuals with severe motor dysfunction (SMD) // Frontiers in Rehabilitation Sciences. – 2022. – URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC9432701/.

6. Hansen D. W., Ji Q. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2010. – Vol. 32, № 3. – P. 478–500. – URL: /mnt/data/af15f6ec-3085-44a0-8752-f302480ccf65.png.

7. Wills S. A., MacKay D. J. C. DASHER — An Efficient Writing System for Brain–Computer Interfaces? // IEEE Transactions on Neural Systems and Rehabilitation Engineering. – 2006. – Vol. 14, № 2. – P. 244–246. – DOI: 10.1109/TNSRE.2006.875573. – URL: https://www.researchgate.net/publication/3430722_DASHER-An_Efficient_Writing_System_for_Brain-Computer_Interfaces/.

8. Rakhmatulin I., Duchowski A. T. Deep Neural Networks for Low-Cost Eye Tracking // Procedia Computer Science. – 2020. – Vol. 176. – P. 685–694. – DOI: 10.1016/j.procs.2020.09.041. – URL: https://www.researchgate.net/publication/344549748_Deep_Neural_Networks_for_Low-Cost_Eye_Tracking/.

9. Paredes-Vallés F., Miyatani Y., Scheper K. Realizing Fully-Integrated, Low-Power, Event-Based Pupil Tracking with Neuromorphic Hardware (preprint). – 2025. – DOI: 10.48550/arXiv.2511.20175. – URL: https://www.researchgate.net/publication/397983835_Realizing_Fully-Integrated_Low-Power_Event-Based_Pupil_Tracking_with_Neuromorphic_Hardware/.

10. Zhang T., et al. Swift-Eye: Towards Anti-blink Pupil Tracking for Precise and Robust High-Frequency Near-Eye Movement Analysis with Event Cameras // IEEE Transactions on Visualization and Computer Graphics. – 2024. – DOI: 10.1109/TVCG.2024.3372039. – URL: https://www.researchgate.net/publication/378711800_Swift-Eye_Towards_Anti-blink_Pupil_Tracking_for_Precise_and_Robust_High-Frequency_Near-Eye_Movement_Analysis_with_Event_Cameras/.

11. Holland C., Komogortsev O. Biometric Identification via Eye Movement Scanpaths in Reading // IEEE International Joint Conference on Biometrics (IJCB). – 2011. – P. 1–8. – DOI: 10.1109/IJCB.2011.6117536. – URL: https://www.researchgate.net/publication/261419713_Biometric_Identification_via_Eye_Movement_Scanpaths_in_Reading/.

12. Goldberg J. H., Kotval X. P. Computer Interface Evaluation Using Eye Movements: Methods and Constructs // International Journal of Industrial Ergonomics. – 1999. – Vol. 24, № 6. – P. 631–645. – DOI: 10.1016/S0169-8141(98)00068-7. – URL: https://www.researchgate.net/publication/222500586_Computer_interface_evaluation_using_eye_movements_Methods_and_constructs/.

13. Cantoni V., Porta M. Eye Tracking as a Computer Input and Interaction Method // Proceedings of the 15th International Conference. – 2014. – P. 1–7. – DOI: 10.1145/2659532.2659592. – URL: https://www.researchgate.net/publication/288492199_Eye_tracking_as_a_computer_input_and_interaction_method/.

14. Ooms K., Krassanakis V. Measuring the Spatial Noise of a Low-Cost Eye Tracker to Enhance Fixation Detection // Journal of Imaging. – 2018. – Vol. 4, № 8. – Article 96. – DOI: 10.3390/jimaging4080096. – URL:

https://www.researchgate.net/publication/326699647_Measuring_the_Spatial_Noise_of_a_Low-Cost_Eye_Tracker_to_Enhance_Fixation_Detection/.

15. Špakov O., et al. Improving the performance of eye trackers with limited spatial accuracy and low sampling rates for reading analysis by heuristic fixation-to-word mapping // Behavior Research Methods. – 2018. – Vol. 51, № 3. – P. 1333–1355. – DOI: 10.3758/s13428-018-1120-x. – URL: https://www.researchgate.net/publication/327705543_Improving_the_performance_of_eye_trackers_with_limited_spatial_accuracy_and_low_sampling_rates_for_reading_analysis_by_heuristic_fixation-to-word_mapping/.

16. Mott M. E., et al. Improving Dwell-Based Gaze Typing with Dynamic, Cascading Dwell Times // Proceedings of the 2017 CHI Conference. – 2017. – P. 1–12. – DOI: 10.1145/3025453.3025517. – URL: https://www.researchgate.net/publication/316653483_Improving_Dwell-Based_Gaze_Typing_with_Dynamic_Cascading_Dwell_Times_/.

17. Sedinkin O., et al. Система для відстеження руху очей на основі машинного навчання // Computer-Integrated Technologies: Education, Science, Production. – 2024. – DOI: 10.36910/6775-2524-0560-2024-55-25. – URL: https://www.researchgate.net/publication/381622618_Sistema_dla_vidstezenna_ruhu_ocej_na_osnovi_masinnogo_navcanna/.

18. Рожков Д. В. Моделювання траєкторії руху цифрових нормативних документів у закладах освіти. – 2021.

19. Агарков Є. С. Дослідження методів проектування AR-інтерфейсів для розробки зручних додатків доповненої реальності. – 2021. – URL: https://openarchive.nure.ua/entities/publication/4693243d-a1f6-4d9d-9eb4-72502c3e0953/.

20. Tokariev V., et al. Ultra Wideband Signals in Control Systems of Unmanned Aerial Vehicles // The 10th IEEE International Conference on Dependable Systems, Services and Technologies (DESSERT'2019), 5–7 June 2019. – Leeds, United Kingdom. – P. 26–29.

21. Tkachov V., et al. Method of Data Collection in Wireless Sensor Networks Using Flying Ad Hoc Network // 5th International Scientific-Practical Conference “Problems of Infocommunications. Science and Technology”, 9–12 October 2018. – Kharkiv, Ukraine. – P. 197–201.

22. Nyström M., Holmqvist K. An adaptive algorithm for fixation, saccade, and glissade detection in eyetracking data // Behavior Research Methods. – 2010. – Vol. 42, № 1. – P. 188–204. – DOI: 10.3758/BRM.42.1.188. – URL: https://www.researchgate.net/publication/41452096/.

23. Holmqvist K., et al. Eye Tracking: A Comprehensive Guide to Methods and Measures. – 2011. – URL: https://www.researchgate.net/publication/254913339_Eye_Tracking_A_Comprehensive_Guide_To_Methods_And_Measures/.

24. Tokariev V., et al. Problem of self-organization of s-bot group movement in unorganized physical environment // Комп’ютерні та інформаційні системи і технології: тези доповідей III міжнар. наук.-техн. конф., 23–24 квіт. 2019 р. – Харків, Україна. – С. 16–17.

25. Churyumov G., et al. Method for Ensuring Survivability of Flying Ad-hoc Network Based on Structural and Functional Reconfiguration // Selected Papers of the XVIII International Scientific and Practical Conference "Information Technologies and Security" (ITS 2018), 27 November 2018. – Kyiv, Ukraine. – P. 64–76.

26. Wood E., Bulling A. EyeTab: Model-based gaze estimation on unmodified tablet computers // Proceedings of the Symposium on Eye Tracking Research and Applications. – 2014. – P. 1–4. – DOI: 10.1145/2578153.2578185.

27. Namnkany O., et al. Comparing Dwell Time, Pursuits and Gaze Gestures for Gaze Interaction on Handheld Mobile Devices // Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI ’23). – April 2023. – Hamburg, Germany. – DOI: 10.1145/3544548.3580871.

28. Komogortsev O., Khan J. I. Eye Movement Prediction by Kalman Filter with Integrated Linear Horizontal Oculomotor Plant Mechanical Model // ETRA 2008, 26–28 March 2008. – Savannah, Georgia, USA. – DOI: 10.1145/1344471.1344525.

29. ISO 9241-210:2019. Human-centred design for interactive systems. – Geneva: ISO, 2019. – 48 p.

30. Duchowski A. Eye Tracking Methodology: Theory and Practice. – Springer, 2003. – 328 p. – DOI: 10.1007/978-1-4471-3750-4.

31. Heikkilä H., Räihä K.-J. Speed and Accuracy of Gaze Gestures // Journal of Eye Movement Research. – 2009. – Vol. 3, № 2. – DOI: 10.16910/jemr.3.2.1.

32. Carberry J. Toward a Unified Theory of High-Energy Metaphysics: Silly String Theory // Journal of Psychoceramics. – 2008. – Vol. 1, № 1. – DOI: 10.5555/12345678.

33. Marnik J. BlinkMouse – On-Screen Mouse Controlled by Eye Blinks // Information Technologies in Biomedicine. – 2014. – P. 237–248. – (Advances in Intelligent Systems and Computing, Vol. 284). – DOI: 10.1007/978-3-319-06596-0_22.

34. Chiang K.-J., et al. A Closed-loop Adaptive Brain-Computer Interface Framework // International IEEE/EMBS Conference on Neural Engineering. – 2021.

35. Vasiljevic G. A. M., de Miranda L. C. Brain–Computer Interface Games Based on Consumer-Grade EEG Devices: A Systematic Literature Review // International Journal of Human–Computer Interaction. – 2020. – Vol. 36, № 2. – P. 105–142.

36. Liu Y., et al. A Robust Recognition Approach in Eye-Based Dwell-Free Typing // Proceedings of the 2015 International Conference on Progress in Informatics and Computing. – 2015. – P. 5–9.

ДОДАТОК А

Модуль, що забезпечує обробку для ока для визначення напрямку погляду.

```
from __future__ import division
import cv2
```

```
from .pupil import Pupil

class Calibration(object):
    def __init__(self):
        self.nb_frames = 20
        self.thresholds_left = []
        self.thresholds_right = []

    def is_complete(self):
        return len(self.thresholds_left) >= self.nb_frames and len(self.thresholds_right) >= self.nb_frames

    def threshold(self, side):
        if side == 0:
            return int(sum(self.thresholds_left) / len(self.thresholds_left))
        elif side == 1:
            return int(sum(self.thresholds_right) / len(self.thresholds_right))

    @staticmethod
    def iris_size(frame):
        frame = frame[5:-5, 5:-5]
        height, width = frame.shape[:2]
        nb_pixels = height * width
        nb_blacks = nb_pixels - cv2.countNonZero(frame)
        return nb_blacks / nb_pixels

    @staticmethod
    def find_best_threshold(eye_frame):
        average_iris_size = 0.48
        trials = {}
```

```
        for threshold in range(5, 100, 5):
            iris_frame = Pupil.image_processing(eye_frame, threshold)
            trials[threshold] = Calibration.iris_size(iris_frame)

        best_threshold, iris_size = min(trials.items(), key=(lambda p: abs(p[1] - average_iris_size)))
        return best_threshold

    def evaluate(self, eye_frame, side):
        threshold = self.find_best_threshold(eye_frame)

        if side == 0:
            self.thresholds_left.append(threshold)
        elif side == 1:
            self.thresholds_right.append(threshold)
```

```
import math
import numpy as np
import cv2
from .pupil import Pupil

class Eye(object):
```

```
    LEFT_EYE_POINTS = [36, 37, 38, 39, 40, 41]
    RIGHT_EYE_POINTS = [42, 43, 44, 45, 46, 47]

    def __init__(self, original_frame, landmarks, side, calibration):
        self.frame = None
        self.origin = None
        self.center = None
        self.pupil = None

        self._analyze(original_frame, landmarks, side, calibration)

    @staticmethod
    def _middle_point(p1, p2):
        x = int((p1.x + p2.x) / 2)
        y = int((p1.y + p2.y) / 2)
        return (x, y)

    def _isolate(self, frame, landmarks, points):
        region = np.array([(landmarks.part(point).x, landmarks.part(point).y) for point in
points])
        region = region.astype(np.int32)

        height, width = frame.shape[:2]
        black_frame = np.zeros((height, width), np.uint8)
        mask = np.full((height, width), 255, np.uint8)
        cv2.fillPoly(mask, [region], (0, 0, 0))
        eye = cv2.bitwise_not(black_frame, frame.copy(), mask=mask)



        margin = 5
```

```
        min_x = np.min(region[:, 0]) - margin
        max_x = np.max(region[:, 0]) + margin
        min_y = np.min(region[:, 1]) - margin
        max_y = np.max(region[:, 1]) + margin

        self.frame = eye[min_y:max_y, min_x:max_x]
        self.origin = (min_x, min_y)

        height, width = self.frame.shape[:2]
        self.center = (width / 2, height / 2)

    def _blinking_ratio(self, landmarks, points):
        left = (landmarks.part(points[0]).x, landmarks.part(points[0]).y)
        right = (landmarks.part(points[3]).x, landmarks.part(points[3]).y)
        top = self._middle_point(landmarks.part(points[1]), landmarks.part(points[2]))
        bottom = self._middle_point(landmarks.part(points[5]), landmarks.part(points[4]))

        eye_width = math.hypot((left[0] - right[0]), (left[1] - right[1]))
        eye_height = math.hypot((top[0] - bottom[0]), (top[1] - bottom[1]))

        try:
            ratio = eye_width / eye_height
        except ZeroDivisionError:
            ratio = None

        return ratio

    def _analyze(self, original_frame, landmarks, side, calibration):
            if side == 0:
            points = self.LEFT_EYE_POINTS
```

```
        elif side == 1:
            points = self.RIGHT_EYE_POINTS
        else:
            return

        self.blinking = self._blinking_ratio(landmarks, points)
        self._isolate(original_frame, landmarks, points)

        if not calibration.is_complete():
            calibration.evaluate(self.frame, side)

        threshold = calibration.threshold(side)
        self.pupil = Pupil(self.frame, threshold)
```

```
import numpy as np
import cv2

class Pupil(object):
    def __init__(self, eye_frame, threshold):
        self.iris_frame = None
```

```
        self.threshold = threshold
        self.x = None
        self.y = None

        self.detect_iris(eye_frame)

    @staticmethod
    def image_processing(eye_frame, threshold):
        kernel = np.ones((3, 3), np.uint8)
        new_frame = cv2.bilateralFilter(eye_frame, 10, 15, 15)
        new_frame = cv2.erode(new_frame, kernel, iterations=3)
        new_frame = cv2.threshold(new_frame, threshold, 255, cv2.THRESH_BINARY)[1]

        return new_frame

    def detect_iris(self, eye_frame):
        self.iris_frame = self.image_processing(eye_frame, self.threshold)

        contours, _ = cv2.findContours(self.iris_frame, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)
        contours = sorted(contours, key=cv2.contourArea)

        try:
            moments = cv2.moments(contours[-2])
            self.x = int(moments['m10'] / moments['m00'])
            self.y = int(moments['m01'] / moments['m00'])
        except (IndexError, ZeroDivisionError):
            pass
```

```
from __future__ import division

import os
import cv2
import dlib
from .eye import Eye
```

```
from .calibration import Calibration

class GazeTracking(object):
    def __init__(self):
        self.frame = None
        self.eye_left = None
        self.eye_right = None
        self.calibration = Calibration()
        self._face_detector = dlib.get_frontal_face_detector()
        cwd = os.path.abspath(os.path.dirname(__file__))
        model_path = os.path.abspath(os.path.join(cwd, "face_landmarks.dat"))
        self._predictor = dlib.shape_predictor(model_path)

    @property
    def pupils_located(self):
        try:
            int(self.eye_left.pupil.x)
            int(self.eye_left.pupil.y)
            int(self.eye_right.pupil.x)
            int(self.eye_right.pupil.y)
            return True
        except Exception:
            return False

    def _analyze(self):
        frame = cv2.cvtColor(self.frame, cv2.COLOR_BGR2GRAY)
        faces = self._face_detector(frame)

        try:
            landmarks = self._predictor(frame, faces[0])
```

```
            self.eye_left = Eye(frame, landmarks, 0, self.calibration)
            self.eye_right = Eye(frame, landmarks, 1, self.calibration)

        except IndexError:
            self.eye_left = None
            self.eye_right = None

    def refresh(self, frame):
        self.frame = frame
        self._analyze()

    def frame_left_coords(self, frame):
            gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            detector = dlib.get_frontal_face_detector()
            cwd = os.path.abspath(os.path.dirname(__file__))
            model_path=os.path.abspath(os.path.join(cwd,"face_landmarks.dat"))
        predictor = dlib.shape_predictor(model_path)
        faces = detector(gray)
        facial_landmarks = predictor(gray, faces[0])
        eye_points=([36, 37, 38, 39, 40, 41])
            left_point=(facial_landmarks.part(eye_points[0]).x,facial_landmarks.part(eye_point
s[0]).y)
            right_point=(facial_landmarks.part(eye_points[3]).x,facial_landmarks.part(eye_poi
nts[3]).y)

center_top=((facial_landmarks.part(eye_points[1]).x+facial_landmarks.part(eye_points[2]
).x)/2, (facial_landmarks.part(eye_points[1]).y+facial_landmarks.part(eye_points[2]).y)/2)
```

```
center_bottom=((facial_landmarks.part(eye_points[5]).x+facial_landmarks.part(eye_point
s[4]).x)/2,(facial_landmarks.part(eye_points[5]).y+facial_landmarks.part(eye_points[4]).y)
/2)
 return ((center_top[0]+center_bottom[0])/2,(center_top[1]+center_bottom[1])/2)

    def pupil_left_coords(self):
        if self.pupils_located:
            x = self.eye_left.origin[0] + self.eye_left.pupil.x
            y = self.eye_left.origin[1] + self.eye_left.pupil.y
            return (x, y)

    def pupil_right_coords(self):
        if self.pupils_located:
            x = self.eye_right.origin[0] + self.eye_left.pupil.x
            y = self.eye_right.origin[1] + self.eye_left.pupil.y
            return (x, y)

    def annotated_frame(self):
        frame = self.frame.copy()

        if self.pupils_located:
            color = (0, 255, 0)
            x_left, y_left = self.pupil_left_coords()
            x_right, y_right = self.pupil_right_coords()
            cv2.line(frame, (x_left - 5, y_left), (x_left + 5, y_left), color)
            cv2.line(frame, (x_left, y_left - 5), (x_left, y_left + 5), color)
            cv2.line(frame, (x_right - 5, y_right), (x_right + 5, y_right), color)
            cv2.line(frame, (x_right, y_right - 5), (x_right, y_right + 5), color)

        return frame
```