

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”**

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

До захисту допущено:

Завідувач кафедри

Сергій СТИРЕНКО

(підпис)

“ ” 2022 р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою “Комп’ютерні системи та мережі”

спеціальності 123 “Комп’ютерна інженерія”

на тему: Система дистанційного транслявання зображення комп’ютера

Виконав : студент 4 курсу, групи ІО-81
(шифр групи)

Сорожинський Михайло Максимович

(прізвище, ім’я, по батькові)

(підпис)

Керівник асистент, Калюжний О.О.

(посада, науковий ступінь, вчене звання, прізвище та ініціали)

(підпис)

Консультант (нормоконтроль) професор, д.т.н., Сімоненко В. П.

(назва розділу)

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

(підпис)

(посада, науковий ступінь, вчене звання, прізвище та ініціали)

(підпис)

Засвідчую, що у цьому дипломному
проєкті немає запозичень з праць інших
авторів без відповідних посилань.

Студент

(підпис)

Київ – 2022 р.

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”**

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Рівень вищої освіти – перший (бакалавр)

Освітньо-професійна програма

“Інженерія програмного забезпечення комп’ютерних систем”

спеціальності 123 “Комп’ютерна інженерія”

ЗАТВЕРДЖУЮ

Завідувач кафедри

Сергій СТИРЕНКО

(підпис)

“ ” 2022 р.

ЗАВДАННЯ

на бакалаврський дипломний проєкт студента

Сорожинського Михайла Максимовича

1. Тема проєкту Система дистанційного транслявання зображення комп’ютера
керівник проєкту Калюжний О.О. асистент,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від “ ” 20__ р. №
2. Термін здачі студентом закінченого проєкту _____
3. Вихідні дані до проєкту технічна документація, теоретичні дані.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які розробляються)
Розділ 1. Огляд потокової передачі даних в інтернеті.
Розділ 2. Розгляд технології AR та способів її реалізації.
Розділ 3. Деталі розробки програми.

5. Перелік графічного матеріалу (з точним позначенням обов'язкових креслень) структурна схема системи, функціональна схема (діаграма класів), алгоритм дій програмного забезпечення.

6. Консультанта проекту, з вказівкою розділів проекту, які до них вносяться

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Нормоконтроль	Сімоненко В. П.		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів проекту	Примітка
1.	Затвердження теми роботи	06.04.2022-07.04.2022	
2.	Вивчення та аналіз завдання	07.04.2022-13.05.2022	
3.	Написання вступної частини та огляду предметної області	13.05.2022-23.05.2022	
4.	Розробка архітектури системи	23.05.2022-01.06.2022	
5.	Програмна реалізація системи	01.06.2022-06.06.2022	
6.	Виправлення помилок	06.06.2022-08.06.2022	
7.	Оформлення документації дипломної роботи	08.06.2022-10.06.2022	
8.	Передзахист	11.06.2022	
9.	Захист	20.06.2022	

Студент-дипломник _____ Михайло СОРОЖИНСЬКИЙ
(підпис)

Керівник проекту _____ Олександр КАЛЮЖНИЙ
(підпис)

АНОТАЦІЯ

В бакалаврському дипломному проєкті реалізовано потоковий спосіб передачі зображення комп'ютера який було модифіковано для пристроїв доповненої реальності .

Програма дозволяє транслявати екран компютера (сервера) у реальному часі та конкатинує передану трансляцію із зображення з камери девайсу (приймача). Програмний продукт був створений на мові програмування Python 3.9, у візуальному середовищі Visual Studio.

Для роботи з зображенням камери девайсу використовується модуль OpenCV 4.5.5., що дозволяє отримати результат виведення камери та використовувати кожний фрейм як окреме зображення для редагування його. Для математичної конкатенації зображень використовується математична логіка пакету NumPy

ANNOTATION

The bachelor's degree project implements a method of transmitting computer images modified for augmented reality devices. The program allows you to broadcast the computer screen (server) in real time and connects the transmitted broadcast from the image from the device's camera (receiver). The software product was created in the Python 3.9 programming language, in the Visual Studio visual environment. To work with the camera image of the device, the OpenCV 4.5.5 module is used, which allows you to get the result of the camera output and use each frame as a separate image to edit it. The mathematical logic of the NumPy package is used for mathematical concatenation of images

ТЕХНІЧНЕ ЗАВДАННЯ
ДО ДИПЛОМНОГО ПРОЄКТУ

на тему: «Система дистанційного транслявання зображення комп'ютера»

ЗМІСТ

НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ.....	2
ПІДСТАВИ ДЛЯ РОЗРОБКИ.....	2
МЕТА ТА ПРИЗНАЧЕННЯ РОЗРОБКИ.....	2
ДЖЕРЕЛА РОЗРОБКИ.....	2
ТЕХНІЧНІ ВИМОГИ.....	3
Вимоги до розробленого продукту.....	3
Вимоги до програмного забезпечення.....	3
Вимоги до апаратної частини.....	3
ЕТАПИ РОЗРОБКИ.....	3

					ІАЛЦ.467100.002 ТЗ								
		№ докум.	Підпис	Дата									
Розробив		Сорожинський М.М			Система дистанційного транслявання зображення комп'ютера Технічне завдання				Літ.	Аркуш	Аркушів		
Перевірив		Калюжний О.О.									1	3	
Н. Контр.		Сімоненко В.											
Затвердив													
					НТУУ КПІ ім. Ігоря Сікорського, ФІОТ,ІО-81								

1 НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Дане технічне завдання поширюється на розробку системи потокової передачі даних та модифікації виводу шляхом доповненої реальності.

Областю застосування цієї системи є продукти доповненої реальності, дизайнерські програми, програми дистанційного керування та області розробки 3D контенту.

2 ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки даної системи є завдання для виконання роботи кваліфікаційно-освітнього рівня «бакалавр інженерії програмного забезпечення», який був затверджений факультетом “Інформатики та обчислювальної техніки” кафедрою обчислювальної техніки Національного технічного Університету України «Київський Політехнічний інститут ім. Ігоря Сікорського».

3 МЕТА ТА ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою та призначенням даної роботи є розробка системи трансляції зображення екрану комп’ютера та конкатинує трансляване зображення з зображенням з камери у реальному часі тим самим створюючи ефект доповненої реальності.

4 ДЖЕРЕЛА РОЗРОБКИ

Джерелом розробки даного дипломного проекту є офіційні документації, публікації та статті в мережі Інтернет на дану тему, науково-технічна література.

					ІАЛЦ.467100.002 ТЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		2

5 ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до розробленого продукту

Розроблена система має виконувати такі вимоги:

- Простий і інтуїтивно-зрозумілий інтерфейс системи.
- Надати користувачам віддалений доступ до девайсу
- Розробити алгоритм накладення зображення для відображення трансляції у доповненій реальності.
- Надати вичерпну та зрозумілу документацію для розробленого продукту.

5.2. Вимоги до програмного забезпечення

- ОС Windows, Mac чи Linux.
- Visual Studio 2021 IDE версії або вище.

5.3. Вимоги до апаратної частини

- ЦП не менше ніж Intel® Core (TM) i3-2100T.
- ROM не менше ніж 64 ГБ.
- RAM не менше ніж 4 ГБ.

6 ЕТАПИ РОЗРОБКИ

Назва етапів виконання	Термін виконання
Затвердження теми роботи	10.12.2020-15.12.2020
Вивчення та аналіз завдання	15.12.2020-15.03.2021
Розробка архітектури та загальної структури системи	15.03.2021-25.03.2021
Розробка структур окремих частин системи	25.03.2021-5.04.2021
Програмна реалізація системи	5.04.2021-15.04.2021
Виправлення помилок	15.04.2021-20.05.2021
Оформлення пояснювальної записки	25.04.2021

					ІАЛЦ.467100.002 ТЗ	Арк.
						3
Зм.	Арк.	№ докум.	Підпис	Дата		

**ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО ДИПЛОМНОГО ПРОЄКТУ**

на тему: «Система дистанційного транслявання зображення комп'ютера»

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ.....	3
ВСТУП.....	4
РОЗДІЛ 1. ОГЛЯД ПОТОКОВОЇ ПЕРЕДАЧІ ДАННИХ В ІНТЕРЕНЕТІ.....	6
1.1 Принцип роботи потокової передачі даних.....	6
1.1.1 Різниця між потоковою передачею даних та завантаженням.....	8
1.1.2 Перевага потокової передачі даних.....	8
1.1.3 Приклади використання потокової передачі даних.....	9
1.1.4 Можливі проблеми при використанні поточкових даних.....	11
1.2 Огляд потокової передачі даних за призначенням.....	11
1.2.1 Передача записаного відео.....	13
1.2.2 Передача даних у живому часі без зворотнього зв'язку.....	14
1.2.3 Передача даних у живому часі зі зворотним зв'язком.....	16
1.3 Розгляд протоколів для потокової передачі даних.....	17
1.3.1 Протокол TCP.....	18
1.3.2 Протокол UDP.....	20
1.3.3 Протокол RTMP.....	21
1.3.4 Протокол RTSP.....	22
1.3.5 Протокол SRT.....	22
1.3.6 Протокол RDP.....	23
ВИСНОВОК ДО РОЗДІЛУ 1.....	24
РОЗДІЛ 2. РОЗГЛЯД ТЕХНОЛОГІЇ AR ТА СПОСОБІВ ЇЇ РЕАЛІЗАЦІЇ.....	25
2.1 Огляд технології AR.....	25
2.1.1 Розгляд роботи AR.....	26
2.1.2 Ключові технології доповненої реальності.....	27
2.1.3 Порівняння AR з подібними технологіями.....	29
2.1.4 Способи використання AR у сучасному житті.....	30

					ІАЛЦ.467100.003 ПЗ			
Зм.	Арк.	№ докум.	Підпис	Дата				
Розробив		Сорожинський М.М			Система дистанційного транслявання комп'ютера Пояснювальна записка	Літ.	Аркуш	Аркушів
Перевірив		Калюжний О.О.					1	106
Реценз.						НТУУ КПІ ім. Ігоря Сікорського, ФІОТ, ІО-81		
Н. Контр.		Сімоненко В.П.						
Затвердив								

2.2 Сучасні способи реалізації AR.....	31
2.2.1 Графічні редактори для створення AR контенту.....	33
2.2.2 Програмне накладання графічного контенту для створення AR	35
2.3 Матричні коди.....	38
2.3.1 Матричний код QR.....	39
2.3.2 Мітка ArUco.....	40
ВИСНОВОК ДО РОЗДІЛУ 2.....	42
РОЗДІЛ 3 ДЕТАЛІ РОЗРОБКИ ПРОГРАМИ.....	44
3.1 Розробка передачі живого потокового зображення.....	44
3.1.1 Клас StreamingClient.....	44
3.1.2 Клас ScreenShareClient.....	47
3.1.3 Клас StreamingServer.....	48
3.2 Створення AR способом конкатинації зображень.....	50
3.2.1 Робота з маркерами ArUco.....	51
3.2.2 Алгоритм накладання зображень.....	53
ВИСНОВОК ДО РОЗДІЛУ 3.....	56
ВИСНОВКИ.....	58
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	59

ПЕРЕЛІК СКОРОЧЕНЬ

LTE	(Long-Term Evolution) Довготерміновий розвиток назва мобільного протоколу передавання даних
ОС	Операційна система
AR	(Argumented Reality) Доповнена реальність
HTTP	(Hypertext Transfer Protocol) Протокол передачі даних
IP	(Internet Protocol) Протокол мережевого рівня для передавання даних між мережами
VR	(Virtual Reality) Віртуальна реальність
MPEG	(Moving Picture Experts Group) Вихідний стандарт відео й аудіо компресії
AAC	(Advanced Audio Coding) Стандартна схема стиснення із втратами для аудіоданих

					ІАЛЦ.467100.003 ПЗ	Арк.
						3
Зм.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Останні роки сучасні комп'ютерні компоненти коштують все дорожче та навіть деякі з них знаходяться у дефіциті, наприклад - відеокарти. У час коли технології затарівають занадто швидко, а нові коштують занадто багато - необхідне альтернативне вирішення проблеми співвідношення потужності до ціни. Замість того щоб купувати додатковий жосткий диск, можна зберігати файли у хмарному сховищі, замість того щоб запускати програму використовуючи свої комп'ютерні потужності, зараз можна запустити програму на дистанційному сервері. Це називається Cloud Computing (Хмарні обчислення). Все більше компаній пропонують свої продукти для дистанційного доступу з хмари заради зниження обчислювальних вимог девайсу користувача.

У нас час ми маємо розвиток бездротових високошвидкісних каналів передачі даних. Зараз найбільш розповсюдженим протоколом передачі даних є 4G LTE. Та зараз набуває розвитку вже п'яте покоління бездротових систем мобільного зв'язку що дозволить передавати файли зі швидкістю 1.056 Гбіт/с.

Тож рано чи пізно можна буде просто транлювати свій комп'ютер дистанційно без будь-яких проблем з затримкою передачі зображення. Крім того за допомогою транслювання зображення можна дати нове життя старим, слабшим девайсам запускаючи на них дистанційні програми чи навіть дистанційні ОС. Тож девайси, що ретранслюють дистанційне зображення хмарних обчислень можуть бути набагато меншими ніж сучасні і потребувати набагато менш потужного процесору, відеокарти та оперативної пам'яті, адже всі обчислення відбуваються дистанційно.

У багатьох компаній на меті створювати більш персональні та зручні девайси. Один з таких девайсів це окуляри доповненої реальності. Компанія Google намагалася зробити саме такі окуляри та у цьому році Xiaomi

					ІАЛЦ.467100.003 ПЗ	Арк.
						4
Зм.	Арк.	№ докум.	Підпис	Дата		

анонсувала Xiaomi Smart Glasses. Тож ми бачимо що намагання компаній створити подібний продукт є доволі великими. Та технологія AR живе не лише в окулярах. Зараз є багато програм що використовують доповнену реальність через камеру смартфона и ноутбку. Тож вона лише чекає достатнього розвитку технологій для того аби показати як сильно зміниться світ з доповненою реальністю.

Тож необхідність у модифікації транслявання зображення для AR незаперечна.

					ІАЛЦ.467100.003 ПЗ	Арк.
						5
Зм.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. ОГЛЯД ПОТОКОВОЇ ПЕРЕДАЧІ ДАННИХ В ІНТЕРЕНЕТІ

1.1 Принцип роботи потокової передачі даних

Перші веб-сайти були простими сторінками з текстом, можливо, із зображенням або двома. Однак сьогодні кожен, хто має достатньо швидке підключення до Інтернету, може дивитися фільми високої чіткості або здійснювати відеодзвінки через Інтернет. Це можливо завдяки технології, яка називається *streaming* (Потокове передавання зображення).

Потокове передавання зображення — це безперервна передача аудіо- або відеофайлів від сервера до клієнта. Простіше кажучи, потокова передача – це те, що відбувається, коли споживачі дивляться телевізор або слухають подкасти на пристроях, підключених до Інтернету. Під час потокового передавання мультимедійний файл, який відтворюється на клієнтському пристрої, зберігається віддалено та передається по кілька секунд через Інтернет.

З поточним відео невід’ємне поняття буферизація. Буферизація — це процес завантаження частини потоку достроково, щоб користувач міг безперервно дивитися відео або слухати аудіо, навіть якщо з’єднання короткочасне. Ви можете побачити це, коли відкриваєте відео YouTube, і панель завантаження в програвачі показує світло-сіру область до кількох секунд після точки відтворення, в якій ви перебуваєте у відео. Буферизація корисна в ситуаціях, коли швидкість мережі низька або існує ймовірність відключення.

Цифрові відеофайли містять значні обсяги даних. Це означає, що вони, як правило, передаються в стиснутому вигляді, щоб завантаження даних використовувало якомога меншу пропускну здатність. Тому програмне забезпечення вашого комп’ютера має декодувати поточні дані, перш ніж

					ІАЛЦ.467100.003 ПЗ	Арк.
						6
Зм.	Арк.	№ докум.	Підпис	Дата		

користувачі зможуть переглядати їх під час відтворення. Якщо користувач переглядає відео YouTube у веб-браузері, наприклад Internet Explorer або Firefox, цей процес декодування зазвичай виконується за допомогою плагінів медіаплеєра, які запущені в програмі браузера. Таким чином, програмне забезпечення має виконувати функції завантаження, декодування та відтворення одночасно, коли користувач переглядає потокове відео.

Потокове передавання працює шляхом розбиття пакетів даних, які становлять відео- або аудіодані, і інтерпретації кожного для відтворення як відео або аудіо в програвачі на пристрої користувача. Це відрізняється від того, що відбувалося до потокового передавання, коли аудіо- або відеофайл потрібно було повністю завантажити на пристрій користувача, перш ніж його можна було відтворити. Хоча це було прийнятним у перші дні Інтернету, коли веб-контент становив лише прості сторінки тексту та статичних зображень, сьогодні ситуація набагато інша.

Високошвидкісне підключення до Інтернету дозволило будь-кому в Інтернеті створювати великі обсяги високоякісного відео та аудіоконтенту, і водночас попит на перегляд такого вмісту також зріс.

Для комфортного перегляду та роботи з трансльованим зображенням необхідна достатня пропускна здатність. Пропускна здатність означає максимальну кількість даних, які можуть передаватися через мережеве з'єднання щосекунди. Це показник того, скільки інформації може пройти через мережу, обчислюється в мегабітах на секунду (Мбіт/с). Як правило, для потокового відео потрібно, щоб ваша мережа мала пропускну здатність щонайменше 4 Мбіт/с, і набагато більше, якщо відео високої чіткості. Пропускна здатність не слід плутати зі швидкістю потокової передачі, яка є мірою того, наскільки швидко можна завантажити вміст. З пропускну здатністю пов'язана концепція затримки або затримки. Затримка — це час, протягом якого інформація надходить до кінцевого користувача. Він також

					ІАЛЦ.467100.003 ПЗ	Арк.
						7
Зм.	Арк.	№ докум.	Підпис	Дата		

відомий як швидкість ring і може сприйматися як затримка, коли ви чекаєте завантаження відео.

1.1.1 Різниця між потоковою передачею даних та завантаженням

Потокова передача в режимі реального часу, це ефективніше, ніж завантаження мультимедійних файлів. Якщо відео завантажено, копія всього файлу зберігається на жорсткому диску пристрою, а відеозапис не може відтворюватися до завершення завантаження всього файлу.

Як і інші дані, що передаються через інтернет, аудіо та відео розбиваються на пакети даних. Кожен пакет містить невеликий фрагмент файлу, аудіо або відеоплеєр у браузері на клієнтському пристрої приймає потік пакетів даних і інтерпретує їх як відео або аудіо.

Потокове передавання за своєю природою відрізняється від завантаження. Коли відео транслюється для відтворення на пристрої користувача, фактичний файл не завантажуються на пристрій. Замість цього пакети відеоданих передаються по кілька разів, так що відео завантажуються частинами, а не за один раз. Коли відео завантажуються на пристрій користувача, копіюється весь файл. Відтворення відео можна почати лише після завантаження всього файлу.

1.1.2 Переваги потокової передачі даних

Обробка поточкових даних є переважною для більшості сценаріїв використання, що передбачають безперервне формування нових динамічних даних. Обробка поточкових даних застосовна у більшості галузевих сегментів і випадків використання, що мають на увазі обробку великих даних. Зазвичай компанії починають із простих завдань, наприклад зі збору даних системних журналів, або з елементарних обчислень, наприклад, з оновлення мінімумів і максимумів. Потім ці завдання трансформуються у складнішу обробку, що відбувається в режимі, близькому до реального часу.

					ІАЛЦ.467100.003 ПЗ	Арк.
						8
Зм.	Арк.	№ докум.	Підпис	Дата		

Спочатку програми можуть обробляти потоки даних з метою формування простих звітів і виконання простих дій у відповідь, наприклад активації сигналізації, коли значення ключових параметрів вийдуть за вказані межі. У результаті цим додаткам доводиться виконувати складніші форми аналізу даних, наприклад застосовувати алгоритми машинного навчання й досягати глибшого розуміння ситуації виходячи з даних. З часом у цей процес також додаються комплексні алгоритми обробки потоків і подій, наприклад аналіз часових вікон для визначення найсвіжіших популярних кінофільмів, що дозволяє отримувати ще кориснішу аналітичну інформацію.

1.1.3 Приклади використання потокової передачі даних

Потокова передача даних зараз використовується майже у кожній галузі, що хоч якось стосується інтернету.

- Датчики, що використовуються в транспортних засобах, промисловому обладнанні та сільськогосподарській техніці, надсилають дані до потокового додатка. Додаток здійснює моніторинг продуктивності, попереджає виникнення можливих дефектів та автоматично замовляє необхідні запасні частини для запобігання простою обладнання.
- Фінансова установа відстежує зміни на фондових біржах у режимі реального часу, обчислює ризикову вартість та автоматично виконує ребалансування портфеля цінних паперів на підставі змін біржового курсу.
- Веб-сайт агентства нерухомості відстежує набір даних, отриманий з мобільних пристроїв клієнтів, та надає рекомендації щодо об'єктів нерухомості в режимі реального часу на підставі даних геолокації.
- Геліоенергетична компанія повинна надавати своїм клієнтам певну прохідну потужність, інакше їй доведеться сплачувати штрафи. Вона розгорнула програму для обробки поточкових даних, яка здійснює моніторинг всіх сонячних батарей і планує необхідне обслуговування в

					ІАЛЦ.467100.003 ПЗ	Арк.
						9
Зм.	Арк.	№ докум.	Підпис	Дата		

режимі реального часу, що дозволяє мінімізувати періоди генерації низької прохідної потужності для кожної з батарей і уникнути сплати штрафів.

- Мультимедійний видавець здійснює потокову передачу мільярдів записів зі своїх онлайн-ресурсів, виконує агрегацію та доповнення даних з урахуванням демографічної інформації про користувачів та оптимізує розміщення контенту на веб-сайті, завдяки чому забезпечується релевантність контенту та підвищується якість обслуговування відвідувачів.
- Компанія-розробник інтернет-ігор виконує збір поточкових даних про взаємодії гравців з іграми та передає ці дані на ігрову платформу. Потім виконується аналіз цих даних у режимі реального часу, внаслідок чого формуються стимули та динамічні ефекти для підвищення залученості гравців.
- Потокове передавання фільмів і телешоу: мабуть, найпоширеніший приклад потокової передачі, це, по суті, функція «відео на вимогу», з провідними платформами потокового медіа, як-от Netflix, Hulu, Disney Plus, Apple TV та інші.
- Пряма трансляція: потокове відео або аудіо вмісту в режимі реального часу, як і коли відбувається подія, через канали соціальних мереж, такі як Facebook, Twitter, Instagram, Tik Tok та інші.
- OTT Streaming: це стосується потокової передачі, коли користувачі отримують доступ до вмісту за допомогою певних пристроїв, таких як настільні комп'ютери, мобільні пристрої, смарт-телевізори або ігрові консолі.
- Пряма трансляція відеоігор: прямі трансляції людей, які грають в ігри, транслюються на пряму аудиторію на таких платформах, як Twitch і YouTube.

					ІАЛЦ.467100.003 ПЗ	Арк.
						10
Зм.	Арк.	№ докум.	Підпис	Дата		

- Музика: аудіофайли, які люди передають, коли хочуть послухати музику – через такі платформи, як Spotify, SoundCloud та Tidal. Подкасти: ще одна форма аудіоформату, яка набирає популярності і яку можна транслювати, щоб люди могли слухати через Apple Podcasts, Stitcher, Spotify та інші програми.

Є також багато інших прикладів використання потокової технології. Будь-яка форма медіаспоживання, коли вміст передається кінцевому користувачеві невеликими пакетами даних замість того, щоб повністю завантажуватися спочатку, може вважатися потоковим.

1.1.4 Можливі проблеми при використанні потокових даних

Обробка поточкових даних вимагає використання двох рівнів: рівня сховища та рівня обробки. Рівень сховища повинен підтримувати черговість записів та сувору несуперечність для забезпечення швидких, економічних та відтворюваних операцій запису та читання великих потоків даних.

Рівень обробки відповідає за споживання даних, розташованих на рівні сховища, виконання обчислень з використанням цих даних та повідомлення рівня сховища про те, які дані можна видалити через непотрібність. Крім того, необхідно передбачити масштабованість, надійність даних та відмовостійкість як на рівні сховища, так і на рівні обробки. В результаті з'явилося безліч платформ, що надають необхідну інфраструктуру для створення програм обробки поточкових даних.

1.2 Огляд потокової передачі даних за призначенням

Під час потокового передавання мультимедійний файл, який відтворюється на клієнтському пристрої, зберігається віддалено та передається по кілька секунд через Інтернет. Тобто потокове передавання зображення через інтернет можна поділити на передачу записаного відео, що

використовують такі сервіси як YouTube, Netflix, AppleTV+ та багато інших, та на сервіси, що транслюють так зване “зображення у живому часі”. Зображення у живому часі це зазвичай трансляція екрану комп'ютера чи зображення камери, що одразу ж передає зображення як тільки його створює чи зчитує.

Транслювання зображення у живому часі поділяються на трансляцію без зворотного зв'язку користувача, тобто коли користувач не може ніяк змінити хід трансляції чи зображення що отримує. Таким способом користуються стримінгові сервіси одним з яких є Twitch. Та стримінг зображення може бути зі зворотним зв'язком.

Ця можливість використовується для дистанційного керування девайсами, наприклад, TeamViewer або спеціальними програмами, наприклад Nvidia GeForce Now, що дозволяє запускати ігри на дистанційному сервері та грати у них на комп'ютерах зі слабкими конфігураціями.



Рис. 1.1 – Способи передачі відео зображення

1.2.1 Передача записаного відео

Більшість стримінгових сервісів використовуються саме для перегляду вже записаного та завантаженого на сервер відео наприклад Netflix або YouTube.

За допомогою потокового передавання ви можете розпочати перегляд відеофайлу без попереднього завантаження повного файлу на свій комп'ютер, як це часто доводиться робити з іншими медіа-об'єктами, такими як зображення. Потокове передавання YouTube дає змогу переглядати різні типи відеофайлів, іноді з різними роздільними можливостями. Розглянемо на прикладі сервісу Youtube.

Перш за все YouTube — американська онлайн-платформа для обміну відео. Його запустили 14 лютого 2005 року Стів Чен, Чад Херлі та Джавед Карім. Він належить Google і є другим за відвідуваністю веб-сайтом після Пошуку Google. YouTube має понад мільярд користувачів щомісяця, які разом переглядають понад мільярд годин відео щодня. Станом на травень 2019 року відео завантажувалися зі швидкістю понад 500 годин вмісту на хвилину.

YouTube в основному використовує відеокодеки VP9 і H.264/MPEG-4 AVC, а також протокол Dynamic Adaptive Streaming через HTTP. Потоки MPEG-4 Частина 2, що містяться в контейнерах 3GP, також надаються для з'єднань з низькою пропускну здатністю. У січні 2019 року YouTube почав випускати відео у форматі AV1. У 2021 році повідомлялося, що компанія розглядає можливість використання AV1 для потокового обладнання, щоб зменшити пропускну здатність і підвищити якість. Відео зазвичай транслюється разом із аудіокодеками Opus і AAC[15].

Користувачі Інтернету можуть транслювати та переглядати відео через YouTube. YouTube розміщує відеофайли на своїх серверах, надсилаючи відеопотоки користувачам, які переходять на сторінку з відео. Користувач може почати перегляд відео майже відразу після того, як він перейде на сторінку, залежно від швидкості його інтернет-з'єднання.

					ІАЛЦ.467100.003 ПЗ	Арк.
						13
Зм.	Арк.	№ докум.	Підпис	Дата		

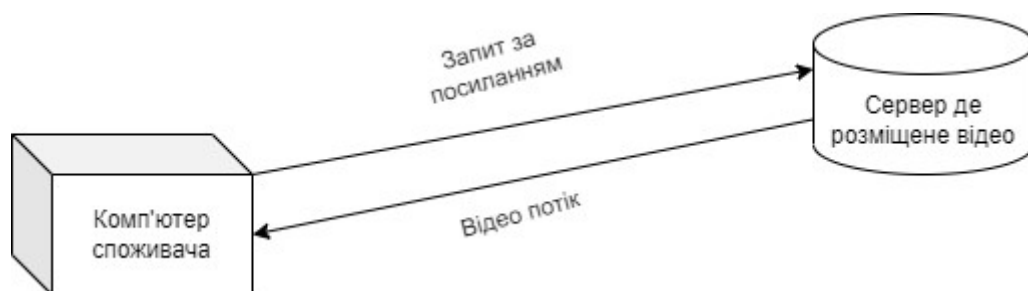


Рис. 1.2 Потокова передача відео YouTube

1.2.2 Передача даних у живому часі без зворотнього зв'язку

Пряма трансляція працює шляхом доставки відео глядачам за допомогою різних протоколів потокової передачі через Інтернет в режимі реального часу та без затримок відео, при цьому HLS, WebRTC і RTMP є найбільш широко використовуваними засобами доставки. Прямі трансляції використовуються для багатьох цілей у різних галузях. По суті, потоковий контент покликаний допомогти людям відвідувати події, виставки та події, які вони не можуть відвідати особисто. Компанії та інші організації використовують прямі трансляції для взаємодії зі своєю аудиторією.

Наприклад такі трансляції використовуються для: віртуальних подій, онлайн-освіти (лекції, тренінги тощо), спортивних подій, трансляції концерту, відеопродажів (демонстрації продуктів, рекламні презентації тощо), стримінгу ігор та інших подій, що відбуваються на екрані монітору стримера.

Найбільш популярним сервісом для такого стримінгу є Twitch. Twitch – це платформа для прямих трансляцій, орієнтована на геймерів, яка була запущена в 2011 році. З тих пір вона зібрала мільйони користувачів, а станом на лютий 2020 року 3,8 мільйона унікальних мовників. Ви можете знайти потік, переглядаючи різні категорії, включаючи конкретні ігри. Якщо ви знайшли стримера, який вам подобається, ви можете стежити за його каналом і отримувати оновлення активності та сповіщення. Та жива потокова передача зображення працює інакше ніж записаного відео. Для збереження або потокової передачі кінцевого вихідного сигналу відеосигнал необхідно

					ІАЛЦ.467100.003 ПЗ	Арк.
						14
Зм.	Арк.	№ докум.	Підпис	Дата		

стиснути через кодер у кодек. Кодеки бувають усіх форм і розмірів. Сам кодер може бути вбудований у камеру, але він також може бути окремим пристроєм, комп'ютерним програмним забезпеченням або мобільним додатком. У багатьох випадках він вбудовується в сучасне програмне забезпечення для мовлення.

Після стиснення відеопотоку через кодер відеодані все ще потрібно запакувати в протокол передачі в реальному часі (RTP) для доставки через Інтернет. Як великий потік безперервних даних, його потрібно розбити на легкозасвоювані шматки або інформаційні пакети. Ці пакети пізніше будуть повторно об'єднані та декодовані в безперервний сигнал медіа-плеєром.

Після упаковки потік або витягується, або відправляється на медіа-сервер. Медіа-сервер може бути в тій же системі, що й кодер, або віддалений десь у хмарі.

Медіа-сервери отримують потік із кодера та конвертують відео за потреби. Вхід – це кількість даних, що надходять на медіа-сервер. Медіасервери можуть перекодувати дані в новий кодек. Вони можуть переводити відео в різні роздільні здатності і навіть транслявати дані, створюючи різний бітрейт. Медіа-сервери також можуть перепакувати або передати дані в різні протоколи. Дані, що передаються з медіа-сервера, відомі як вихід[14].

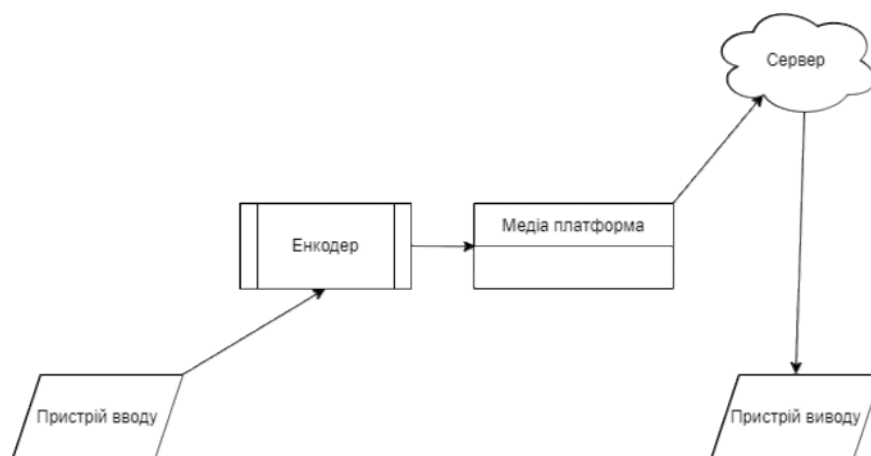


Рис. 1.3 Потокова передача відео Twitch

1.2.3 Передача даних у живому часі зі зворотнім зв'язком

Зворотній зв'язок під час живої потокової трансляції використовуються для потреб керування тим зображенням, що передається, в більшості для керування віддаленими девайсами. Основним прикладом віддаленого робочого столу є підключення домашнього ноутбука до офісного ПК, щоб ви могли отримувати доступ до файлів, запускати програми, друкувати документи тощо на цьому ПК, не заходячи в офіс. Однак хост-пристрій не обов'язково повинен бути ПК. Часто це серверне або віртуальне серверне середовище в багатьох випадках використання бізнесу.

За допомогою віддаленого робочого столу хост-пристрій забезпечує живлення програмного забезпечення та операційної системи та відображає їх на клієнтському пристрої. Програмне забезпечення віддаленого робочого столу фіксує введення миші та клавіатури з клієнтського пристрою та надсилає їх назад на хост-комп'ютер. Для завдань і файлів, які вимагають інтенсивного використання графіки, може знадобитися додаткове програмне забезпечення для забезпечення плавної та швидкої роботи. Однією з найпопулярніших сервісів для таких потреб є TeamViewer.

Хмарна платформа для віддалених підключень TeamViewer забезпечує безпечний віддалений доступ до будь-якого пристрою незалежно від платформи у будь-який час та з будь-якого місця. TeamViewer можна використовувати для підключення до комп'ютерів, смартфонів, серверів, пристроїв Інтернету речей, робот тощо. Швидкі та ефективні з'єднання за допомогою глобальної мережі доступу, навіть, у зовнішніх просторах або в зонах зі слабкою смугою пропускання.

Teamviewer використовує порт 80 для підключення до центрального сервера. Якщо з'єднання встановлено, ви отримаєте унікальний ідентифікатор, і сервер дізнається, що ви в мережі. Весь зв'язок може відбуватися через порт 80, якщо інші порти заблоковані. Teamviewer дозволяє підключатися безпосередньо до IP-адреси. Під час встановлення сеансу TeamViewer визначає

					ІАЛЦ.467100.003 ПЗ	Арк.
						16
Зм.	Арк.	№ докум.	Підпис	Дата		

оптимальний тип підключення. Після “рукостискання” через головні сервери пряме з’єднання через UDP або TCP встановлюється в 70% усіх випадків (навіть за стандартними шлюзами, NAT і брандмауерами). Решта з’єднань маршрутизуються через мережу маршрутизатора з великим надлишком через TCP або https-тунелювання.



Рис. 1.4 Потокова передача відео TeamViewer

1.3 Розгляд протоколів для потокової передачі даних

Кожен з попередніх способів використання потокової передачі даних використовують різні протоколи.

Для передачі записаного відео деякі методи потокової передачі використовують UDP, а деякі використовують TCP. UDP і TCP є транспортними протоколами, тобто вони використовуються для переміщення пакетів даних між мережами. Обидва використовуються з Інтернет-протоколом (IP). TCP відкриває спеціальне з’єднання перед передачею даних і гарантує, що всі пакети даних надходять у порядку. На відміну від TCP, UDP не виконує жодної з цих речей. В результаті TCP є надійнішим, але передача даних через UDP не займає стільки часу, як через TCP, хоча деякі пакети втрачаються на цьому шляху.

Для прямох трансляції відео без зворотнього зв’язку використовуються інші протоколи такі як RTSP, SRT або RTMP.

У той час коли для потокової передачі зі зворотним зв'язком часто використовують протокол RDP[3].

1.3.1 Протокол ТСП

Протокол керування передачею (ТСП) орієнтований на з'єднання, тобто після встановлення з'єднання дані можуть передаватися в двох напрямках. ТСП має вбудовані системи для перевірки на наявність помилок і гарантії доставки даних у тому порядку, в якому вони були надіслані, що робить його ідеальним протоколом для передачі такої інформації, як нерухомі зображення, файли даних та веб-сторінки. Але хоча ТСП є інстинктивно надійним, його механізми зворотного зв'язку також призводять до більших витрат, що призводить до більшого використання доступної пропускної здатності у вашій мережі.

Протокол керування передачею (ТСП) орієнтований на з'єднання, тобто після встановлення з'єднання дані можуть передаватися в двох напрямках. ТСП має вбудовані системи для перевірки на наявність помилок і гарантії доставки даних у тому порядку, в якому вони були надіслані, що робить його ідеальним протоколом для передачі такої інформації, як нерухомі зображення, файли даних та веб-сторінки. Але хоча ТСП є інстинктивно надійним, його механізми зворотного зв'язку також призводять до більших витрат, що призводить до більшого використання доступної пропускної здатності у вашій мережі.

Перед тим, як дані можуть бути передані між двома вузлами, ТСП, на відміну від UDP, передбачена стадія установки з'єднання. Також після того, як всі дані були передані, настає стадія завершення з'єднання. Таким чином, здійснення кожного ТСП-з'єднання можна умовно поділити на три фази:

- Ініціалізація з'єднання - налагодження з'єднання здійснюється за допомогою так званого тристороннього рукостискання ТСП. Ініціатором з'єднання може бути будь-яка сторона.

					ІАЛЦ.467100.003 ПЗ	Арк.
						18
Зм.	Арк.	№ докум.	Підпис	Дата		

- Загрузка даних - після ініціалізації з'єднання корисне навантаження переміщатиметься в обох напрямках TCP-з'єднання. Усі пакети обов'язково міститимуть встановлений прапор ACK. Інші прапори, такі як PSH або URG, можуть бути, а можуть і не бути встановленими.
- Завершення з'єднання - При нормальному завершенні TCP-з'єднання в більшості випадків ініціалізується процедура, яка називається двостороннім рукостисканням, під час якої кожна сторона закриває свій кінець віртуального каналу та звільняє всі задіяні ресурси. Зазвичай ця фаза починається з того, що один із задіяних процесів програми сигналізує своєму рівню TCP, що сеанс зв'язку більше не потрібен. З боку цього пристрою надсилається повідомлення з встановленим прапором FIN (зазначимо, що цей пакет не обов'язково повинен бути порожнім, він також може містити корисне навантаження), щоб повідомити іншій пристрій про бажання завершити відкрите з'єднання. Потім отримання цього повідомлення підтверджується (повідомлення від пристрою, що відповідає, з встановленим прапором ACK, що говорить про отримання повідомлення FIN). Коли пристрій готовий, він також надсилає повідомлення з встановленим прапором FIN, і, після отримання у відповідь підтверджує отримання повідомлення з встановленим прапором ACK або очікування певного періоду часу, передбаченого для отримання ACK, сеанс повністю закривається. Стани, через які проходять два з'єднані пристрої під час звичайного завершення з'єднання, відрізняються, тому що пристрій, що ініціює завершення сеансу, поводить трохи інакше, ніж пристрій, який отримує запит на завершення[12].

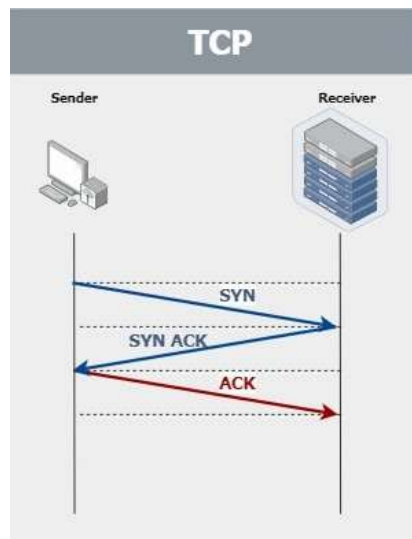


Рис. 1.5 Схема роботи протоколу TCP

1.3.2 Протокол UDP

Протокол дейтаграм користувача (UDP) — це простіший Інтернет-протокол без з'єднання, в якому не потрібні послуги перевірки помилок та відновлення. З UDP немає накладних витрат на відкриття з'єднання, підтримку з'єднання або припинення з'єднання; дані безперервно надсилаються одержувачу, незалежно від того, отримав він їх чи ні. Хоча UDP не ідеально підходить для надсилання електронної пошти, перегляду веб-сторінки чи завантаження файлу, він переважно є кращим для спілкування в реальному часі, як-от трансляція або багатозадачна мережева передача.

Для потокової передачі в деяких випадках швидкість набагато важливіша за надійність. Наприклад, якщо хтось бере участь у відеоконференції, він вважає за краще взаємодіяти з іншими учасниками конференції в режимі реального часу, ніж сидіти і чекати, поки буде доставлено кожен біт даних. Тому кілька втрачених пакетів даних не є великою проблемою, і слід використовувати UDP. В інших випадках надійність важливіша для потокової передачі. Наприклад, як HTTP live streaming (HLS), так і MPEG-DASH є поточковими протоколами, які використовують TCP для транспортування. Багато служб відео на вимогу використовують TCP[11].

Елементи керування потоком від TCP, хоча і є надійними, не здатні відновити відсутні дані досить швидко, щоб бути корисними у відеозв'язку в реальному часі. І хоча цілісність даних важлива, вона має бути збалансована зі швидкістю, щоб забезпечити безперешкодний темп спілкування.

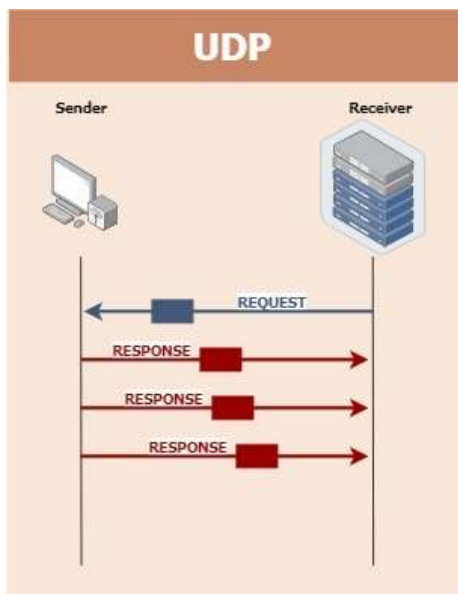


Рис. 1.6 Схема роботи протоколу UDP

1.3.3 Протокол RTSP

RTSP, скорочення від Real-Time Streaming Protocol, — це протокол, структурований подібно до RTMP, який використовується для надсилання команд від користувача до відеопрогравача. Сервери RTSP знаходяться між прямим потоком і глядачем, видають команди «відтворення», «пауза» та «запис». Цей протокол набагато менш популярний, ніж інші, про які ми згадували, але все одно дуже важливий.

Оскільки RTSP розроблено для захисту від втрати аудіо/відео під час проблем із підключенням, мовники можуть бути спокійні, знаючи, що на їх відео не вплинуть тимчасові перебої з'єднання або вузькі місця під час завантаження. Цей захист настільки великий, що RSP навіть підтримує комутацію мереж, наприклад перехід від одного оператора стільникового зв'язку до іншого в середині потоку, без втрати даних. Користуючись

перевагами маршрутизатора Multi-WAN, кодери, які використовують RSP, можуть використовувати кілька підключень до Інтернету одночасно. Ці додаткові мережі можуть використовуватися як резервні або для забезпечення додаткової пропускної здатності для завантаження.

1.3.4 Протокол RTMP

RTMP, що є скороченням від Real-Time Messaging Protocol, був важливим для прямої трансляції, оскільки Flash Player був стандартним відеопрогравачем.

Цей транспортний протокол раніше відповідав за доставку прямої трансляції, але тепер він відповідає за прийом RTMP з кодера. RTMP часто поєднується з доставкою HLS для оптимального налаштування потокової передачі. Ця комбінація забезпечує низьку затримку та надійну безпеку[10].

RTMP визначає кілька віртуальних каналів, по яких пакети можуть надсилатися та прийматися, і які працюють незалежно один від одного. Наприклад, є канал для обробки запитів і відповідей RPC, канал для даних відеопотоку, канал для даних аудіопотоку, канал для позасмугових керуючих повідомлень (узгодження розміру фрагменту тощо) тощо. . Під час типового сеансу RTMP кілька каналів можуть бути активними одночасно в будь-який момент часу. Коли дані RTMP кодуються, генерується заголовок пакета. Заголовок пакета вказує, серед іншого, ідентифікатор каналу, по якому він має бути відправлений, мітку часу, коли він був створений (якщо необхідно), і розмір корисного навантаження пакета.

1.3.5 Протокол SRT

SRT, що є скороченням від Secure Reliable Transport, — це протокол потокової передачі, який допомагає забезпечити безпечне потокове передавання через загальнодоступні мережі. Він має низьку затримку та високобезпечну трансляцію. Цей протокол також є відкритим вихідним кодом,

					ІАЛЦ.467100.003 ПЗ	Арк.
						22
Зм.	Арк.	№ докум.	Підпис	Дата		

що полегшує його впровадження. На даний момент SRT не такий популярний, як HLS і RTMP, оскільки він є відносно новим, а найпопулярніші інструменти мовлення ще не сумісні.

SRT популярний у випадках використання нестабільних або ненадійних мереж. Будь-який робочий процес потокової передачі, ускладнений проблемами першої милі, може отримати користь від цієї технології. Це також чудовий протокол для створення двосторонніх програм і синдикування потоків із кількох джерел[8].

1.3.6 Протокол RDP

Протокол віддаленого робочого столу (RDP) є частиною служб віддаленого робочого столу і це просто те, що Microsoft колись назвала своїм власним протоколом для доставки віддаленого робочого столу через графічний інтерфейс. Наразі Microsoft називає своє офіційне програмне забезпечення клієнта RDP підключенням до віддаленого робочого столу (раніше — клієнт служб терміналів). Кожна версія Windows, починаючи з Windows XP, містить RDC. Версії RDC визначаються версією Windows або останнім застосованим пакетом оновлень Windows[9].

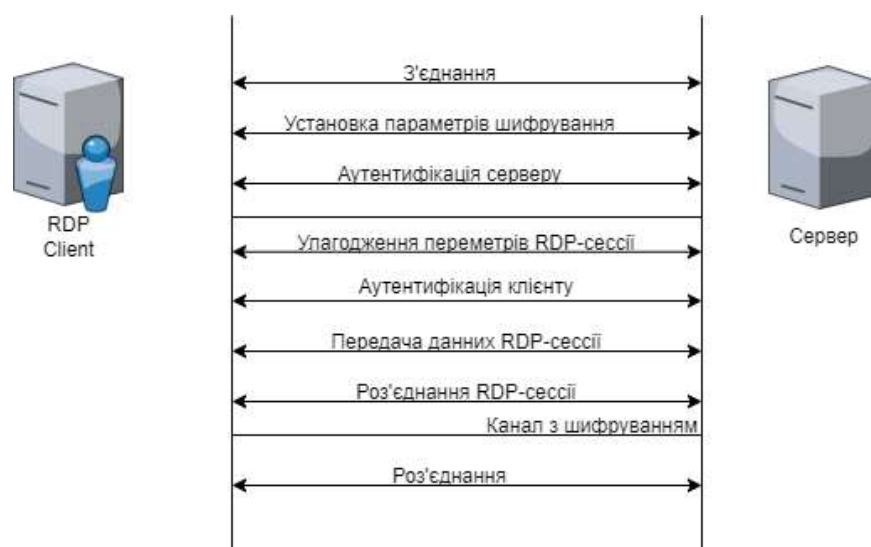


Рис. 1.9 Схема роботи протоколу RDP

ВИСНОВОК ДО РОЗДІЛУ 1

Розглянуто потокову передачу зображення через інтернет, принцип роботи та її класифікацію за метою використання. Виділено її недостатки та переваги та наведені приклади використання данної передачі інформації у різних відомих сервісах.

Розглянуті можливі протоколи, для використання передачі даних в інтрнеті для різних потреб.

Після огляду усього зазначеного ми переконуємося, що потокова передача даних найкращий спосіб реалізації передачі відео у реальному часі та найкраще буде використовувати протокол UDP для передачі зображення та підключення по HTTP IP по 80 порту для віддаленого керування.

					ІАЛЦ.467100.003 ПЗ	Арк.
						24
Зм.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2. РОЗГЛЯД ТЕХНОЛОГІЇ AR ТА СПОСОБІВ ЇЇ РЕАЛІЗАЦІЇ

2.1 Огляд технології AR

Є багато можливостей для дизайнерів і брендів, щоб вивчити можливості доповненої реальності (AR) і те, як вона може покращити творчу роботу, забезпечити розваги або покращити суспільство. Технологічні гіганти, такі як Microsoft, Google і Apple, а також все, від дитячих книжок до 3D-моделювання для ігор, експериментують з AR. Велика кількість безкоштовних додатків для створення контенту демократизує AR, а це означає, що будь-хто (не тільки розробники) може створювати власний досвід AR. І з найближчим випуском першого телефону на ринку, сумісного з AR і VR – Asus Zenfone AR (показано нижче) – ми подивимося, що він робить і що це означає для демократизації дорогих і недоступних на даний момент технологій для більшості громадськості.

Технологія доповненої реальності (AR) — це технологія, яка поєднує віртуальну інформацію з реальною світ. Технічні засоби, які він використовує, включають мультимедіа, 3D-моделювання, відстеження в реальному часі та Реєстрація, інтелектуальна взаємодія, розпізнавання тощо. Його принцип полягає в застосуванні комп'ютерного віртуальну інформацію, таку як текст, зображення, 3D-моделі, музика, відео тощо, у реальний світ після моделювання. Таким чином, два види інформації доповнюють один одного, досягаючи таким чином покращення реальний світ.

Технологія доповненої реальності (AR) — це технологія, яка поєднує віртуальну інформацію з реальною світ. Технічні засоби, які він використовує, включають мультимедіа, 3D-моделювання, відстеження в реальному часі та Реєстрація, інтелектуальна взаємодія, розпізнавання тощо. Його принцип полягає в застосуванні комп'ютерного віртуальну інформацію, таку як текст,

					ІАЛЦ.467100.003 ПЗ	Арк.
						25
Зм.	Арк.	№ докум.	Підпис	Дата		

зображення, 3D-моделі, музика, відео тощо, у реальний світ після моделювання. Таким чином, два види інформації доповнюють один одного, досягаючи таким чином покращення реальний світ.

Інтелектуальна технологія відображення, технологія тривимірної реєстрації та технологія інтелектуальної взаємодії становлять основне технологічне коло AR і відіграють важливу роль у розвитку AR

2.1.1 Розгляд роботи AR

Доповнена реальність (AR) використовує існуючу реальність і фізичні об'єкти, щоб ініціювати створені комп'ютером покращення над реальністю в реальному часі. По суті, AR – це технологія, яка накладає згенеровані комп'ютером зображення на уявлення користувача про реальний світ. Ці зображення зазвичай мають форму 3D-моделей, відео та інформації. Як це накладається, залежить від характеру досвіду та обладнання, на якому ви переглядаєте досвід. Найпростішим способом є використання телефону, де те, що ви бачите через камеру, має цифрові елементи. Найвідомішим прикладом цього є Pokemon Go, але його також використовували, щоб направляти людей по місцях, додаючи віртуальні вказівники, що вказують вказівки, або в художніх галереях, щоб отримати інформацію про твори мистецтва, коли ви дивитесь на них «через телефон»[16].

Доповнена реальність працює в поєднанні з гарнітурами та/або цифровими пристроями, такими як планшети, смартфони і навіть ПК. Самі пристрої містять програмне забезпечення, датчики та цифрові проектори, які запускають цифрові дисплеї на фізичні об'єкти. AR можна створювати і використовувати в різних формах. Наприклад, Google Glass відображає 2D-зображення на прозорих окулярах, тоді як HoloLens від Microsoft вбудовує 3D-зображення у світ навколо вас. Такі додатки, як Blippar, Zappar і Aurasma, дозволяють створювати доступний (якщо не безкоштовний) контент AR для малого бізнесу, освіти та багатьох інших галузей, а також, звичайно, функції

					ІАЛЦ.467100.003 ПЗ	Арк.
						26
Зм.	Арк.	№ докум.	Підпис	Дата		

AR в існуючих програмах та іграх для смартфонів, таких як Snapchat і Pokémon Go. У процесі AR використовується камера з «режимом сканування». Щоб використовувати деякі програми для створення доповненої реальності, такі як Vuforia, вам потрібно навести камеру сканування на різні об'єкти в кімнаті, щоб створити базу даних форм і кутів. Як тільки камера виявить об'єкт, що «запускає» зі своєї бази даних, цифровий об'єкт на екрані з'явиться в положенні цільового об'єкта.

Для розробників цей процес включає в себе масу основних алгоритмів, і іноді системі доводиться одночасно виявляти тисячі різних цільових зображень.

2.1.2 Ключові технології доповненої реальності

Інтелектуальна технологія відображення, технологія тривимірної реєстрації та технологія інтелектуальної взаємодії становлять основне технологічне коло AR і відіграють важливу роль у розвитку AR.

- Інтелектуальна технологія дисплея - згідно з відповідними даними, понад 65% інформації, отриманої людиною, походить з власне бачення, яке стало для людини найбільш інтуїтивним способом взаємодії з реальним навколишнє середовище. З розвитком технології інтелектуального дисплея доповнена реальність стає а можливість, яка висувається на нову висоту завдяки різноманітним видам пристроїв відображення, створених на основі технологія інтелектуального відображення. Зокрема, існує три основні категорії пристроїв відображення займають сьогодні важливе місце в області технологій AR. Спочатку був шоломний дисплей (HMD) 1968 р.н., оптичний перспективний дисплей шолома, розроблений професором Іваном Сазерлендом. можливість накладення простих графічних елементів, створених комп'ютером, на реальні сцени в режимі реального часу. В пізніша розробка, оптична перспектива, встановлена н а шолом, і

					ІАЛЦ.467100.003 ПЗ	Арк.
						27
Зм.	Арк.	№ докум.	Підпис	Дата		

відео перспектива, встановлена на шоломі. По-друге, кишеньковий пристрій відображення, покладаючись на технологію доповненої реальності портативного дисплея дисплей портативного пристрою дуже легкий, маленький, особливо популярність смартфонів, через перспективу відео до використання доповненої реальності представити технологію. По-третє, інші пристрої відображення, такі як настільні дисплеї ПК, відповідають дійсному світу інформацію про сцену, зняту камерою, у тривимірну віртуальну модель, створену за допомогою комп'ютера і в кінцевому підсумку відображаються на дисплеї робочого столу.

- Технологія 3D реєстрації - технологія 3D реєстрації, як одна з найважливіших технологій у системі доповненої реальності дозволяє точно накладати віртуальні зображення в реальне середовище. Основний потік 3d Технологія реєстрації складається з двох етапів. Спочатку визначте співвідношення між віртуальним зображенням модель, а також інформацію про напрям і положення камери або пристрою відображення. По-друге, віртуальний відтворене зображення і модель точно проектується в реальне середовище, тому віртуальне зображення і Модель може бути злита з реальним середовищем. Існують різні способи 3D-реєстрації, наприклад технологія реєстрації на базі апаратного трекера, технологія 3d реєстрації на базі комп'ютера vision, технологія тривимірної реєстрації на основі бездротової мережі та змішана технологія реєстрації, серед яких перші два є найпопулярнішими. Для технології тривимірної реєстрації на основі комп'ютерного зору, він встановлює орієнтир для реалізації визначення напрямку і положення реальної сцени камерою або дисплеєм.
- Технологія інтелектуальної взаємодії - інтелектуальна інтерактивна технологія тісно пов'язана з технологією інтелектуального відображення, 3D-реєстрацією технології, ергономіка, когнітивна

					ІАЛЦ.467100.003 ПЗ	Арк.
						28
Зм.	Арк.	№ докум.	Підпис	Дата		

психологія та інші дисципліни. У системах AR існує різноманітність інтелектуальних взаємодій, включаючи взаємодії з апаратним пристроєм, взаємодію з розташуванням, на основі тегів або інші інформаційні взаємодії. З розвитком технології інтелектуальної взаємодії, доповнена реальність не тільки накладає віртуальну інформацію на реальні сцени, але й реалізує взаємодія людей з віртуальними об'єктами в реальних сценах. Ця взаємодія заснована на тому, що люди дають конкретні інструкції віртуальному об'єкту на сцені, і віртуальний об'єкт може зробити деякі зворотний зв'язок, що дозволить аудиторії програми доповненої реальності отримати кращий досвід.

2.1.3 Порівняння AR з подібними технологіями

Microsoft повернула термін «змішана реальність» (MR) у центр уваги, коли випустила HoloLens (подивіться, що HoloLens може зробити для дизайнерів і художників тут). Деякі припускають, що це могло полягати в тому, щоб виділити їхню технологію як щось нове та відмінне. Microsoft каже, що це було використано для спрощення спектру VR/AR. У будь-якому випадку, новий термін, можливо, спричинив більше плутанини, ніж він того варте.

Змішана реальність дозволяє користувачам взаємодіяти з віртуальними голограмами та реальними об'єктами у своєму фізичному світі. Хоча це те, що робить доповнена реальність – кажуть, що змішана реальність прив'язує віртуальні об'єкти до точки реального простору, що дає змогу розглядати їх як «реальні» з точки зору людини, яка використовує MR-пристрої. По суті, цифрову ілюзію важче зламати. Доповнену реальність також не слід плутати з віртуальною реальністю (VR), яка є повністю захоплюючим досвідом для користувача без взаємодії фізичного світу навколо нього. VR — це створене комп'ютером моделювання 3D-зображень (або зазвичай середовища), з якими

					ІАЛЦ.467100.003 ПЗ	Арк.
						29
Зм.	Арк.	№ докум.	Підпис	Дата		

користувач може взаємодіяти, здавалося б, реально за допомогою гарнітури та/або рукавичок, але, по суті, це досить ізоляційний досвід.

2.1.4 Способи використання AR у сучасному житті

Дослідження того, в яких сферах AR зараз використовується, показало, що можливості безмежні, але регулярна інтеграція ще не відійшла далеко від маркетингу. В останній розробці Apple випустила власний ARKit в iOS 11, що полегшує розробникам пропонувати можливості AR у своїх програмах без необхідності розробляти систему AR, але він доступний лише для дорогих продуктів Apple. Якщо ви хочете дізнатися про розробки Apple AR, прочитайте цей останній звіт про чутки на нашому дочірньому сайті Macworld UK.

Ще одним великим гравцем AR є Google і його основна технологія Project Tango. Пропозиції Google Project Tango (AR) і Daydream (VR) доступні з ASUS ZenFone AR, смартфоном, доступним для попереднього замовлення у Великобританії, дата випуску якого відбудеться в кінці цього місяця.

Це може здатися нецікавим за номінальною вартістю, але це ознака того, що AR і VR можуть бути доступні більшій частині громадськості, а не лише тим, хто може дозволити собі дорогі гарнітури. Хоча смартфон сам по собі недешевий (або особливо відомий бренд у Великобританії), це означає, що світ розваг AR і VR може бути доступним у вас під рукою.

Essentially Project Tango — це набір датчиків і програмного забезпечення комп'ютерного зору, розробленого Google для доповненої реальності. Як і функції інших додатків AR, Zenfone AR відстежує рух, запам'ятовує фізичне оточення і може використовувати цю технологію для створення інтерактивного світу, пов'язаного з вашою реальністю. Але телефон також сумісний з Google Daydream. Це остання мобільна гарнітура Google для перегляду віртуальної реальності – крок уперед від Google Cardboard.

					ІАЛЦ.467100.003 ПЗ	Арк.
						30
Зм.	Арк.	№ докум.	Підпис	Дата		

Використовуючи Asus Zenfone AR, ви можете насолоджуватися різноманітними програмами та іграми Daydream, такими як Netflix VR.

Телефон із розширеною матрицею датчиків вперше був анонсований на CES цього року, але випуск у Великобританії залишився неоднозначним – «це літо» було зазначено найбільш детально. Але тепер ви можете зробити попереднє замовлення в Asus за £799,99 з очікуваним днем випуску з 28 серпня по 1 вересня. Один з наших колег, Льюїс Пейнтер, на нашому дочірньому веб-сайті, Tech Advisor, зумів отримати телефон Asus Zenfone AR у для перегляду. Перевірте, що він думав.

Тим часом Microsoft непомітно показала новий прототип окулярів AR, які виглядають як звичайні окуляри в товстій оправі, припускаючи, що це може бути менш дорогим продуктом для HoloLens, випущеного два роки тому. Очікується, що більше про це буде оголошено на конференції Siggraph у серпні.

Такі галузі, як архітектура, освіта, ігри, кіно, медицина, дитячі книги, військові та навіть скромні аспіранти (див. остаточний проект студента LCC Пітера Родена), експериментують і використовують AR. Нижче ми окреслимо деякі з основних галузей.

2.2 Сучасні способи реалізації AR

Сьогодні можливі два шляхи створення графічного контенту з використанням AR. Один з них це використання готового графічного редактору. Інший це самостійне або програмне накладання графічного контенту на відео у реальному часі.

Технологія AR застосовувалася в багатьох галузях, включаючи туризм, археологію, мистецтво, торгівлю, промислове виробництво та відновлення, освіта, управління надзвичайними ситуаціями, розваги та дозвілля, та медичне лікування. У сфері туризму можна застосувати технологію доповненої

					ІАЛЦ.467100.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		31

реальності до туристичних об'єктів відновлювати історичні місця за допомогою камер мобільного телефону, екранного програмного забезпечення та інших технологічних засобів щоб інтегрувати реальні сцени. Окрім перегляду сцен, можна отримати додаткову інформацію. Технологія AR часто використовується в археологічних дослідженнях, щоб збільшити реліквії в реальних ландшафтах гарантувати, що археологи можуть точніше визначити їх місцезнаходження.

Технологія AR часто використовується в археологічних дослідженнях, щоб збільшити реліквії в реальних ландшафтах гарантувати, що археологи можуть точніше визначити їх місцезнаходження. Технологія AR дозволяє споживачам побачити все про продукт, не відкриваючи його упаковки. Скануючи зображення продукту, ви можете не тільки відобразити деяку інформацію про користувацькі параметри, а й отримати іншу інформацію про зображення продукту. Технологія AR може направляти працівників візуально, нагадувати їм про час і малювати в 3D для створення продуктів більш ефективно. Візьмемо, як приклад автомобільну промисловість, технологія AR може дати можливість дизайнери автомобілів, щоб краще покращити структуру автомобілів і зробити кращі порівняння за допомогою візуальних презентацій.

Застосування технології AR у сфері мистецтва дозволило людям мати більше ракурсів досвід і інтерпретація дійсності. Часто це злиття реальності та реальності перетворюється на форму мистецтва. Технологія AR може використовуватися в рішеннях громадської безпеки і зіграла хорошу роль. Наприклад, Система доповненої реальності для пошуку та порятунку оснащена аерофотокамерою, яка може інтегрувати реальна сцена з назвою лісової дороги та місцем, визначеним географією, щоб врятувати загублену людину більш ефективно.

У найближчі кілька років почнеться застосування технології доповненої реальності, особливо в додатку мобільних інтелектуальних терміналів,

					ІАЛЦ.467100.003 ПЗ	Арк.
						32
Зм.	Арк.	№ докум.	Підпис	Дата		

з'явиться у великій кількості. Хоча мобільних пристроїв менше занурювані дисплеї, ніж дисплеї, встановлені на шолом, вони дуже популярні. У той же час, запуск с Платформи розробки ARKit і ARCore реалізують поєднання доповненої реальності та розумного мобільні пристрої технічно. У майбутньому розробка розумних носимих пристроїв, які можуть дати повну гру до переваг технології доповненої реальності, створить більш реалістичний світ інтеграції для людські істоти. Люди можуть взаємодіяти з системою більш природним способом взаємодії людини та комп'ютера. У майбутньому технологія доповненої реальності значною мірою змінить життя людини, що є неминуча тенденція науково-технічного розвитку.

Лікар може використовувати технологію AR для більш точного визначення місця хірургічного втручання пацієнта. AR технологія дозволяє краще спостерігати за плід у режимі реального часу. Технологія AR також може нагадати пацієнту приймайте ліки вчасно, дозволяючи пацієнту носити відповідне обладнання[4].

2.2.1 Графічні редактори для створення AR контенту

Графічні редактори користуються популярністю через зручність та простоту їх використання. Вони подібну до будь-якого іншого графічного редактору та мають зручний інтерфейс користувача. Тому створення AR доступно усім користувачам хто має можливість використання подібних програм. Розглянемо деякі з них.

- Blippar використовують такі бренди, як Dr Pepper і Marvel Studios, або як освітній інструмент. За допомогою Blippbuilder або Blippbuilder Script кожен може створити свій досвід AR.

Проект Scarlet Raven The Danger Tree є видатним прикладом того, як використовувати AR як художника. Після завантаження програми Blippar необхідно покласти телефон на її традиційні картини, щоб побачити нові сцени та інтерпретації Першої світової війни.

					ІАЛЦ.467100.003 ПЗ	Арк.
						33
Зм.	Арк.	№ докум.	Підпис	Дата		

- Zappar. Іншим постачальником програм і програмного забезпечення AR для «демократизації» технології є Zappar. Не плутати з Google Cardboard і значно дешевше, ніж HoloLens від Microsoft, Zappar займає цікаве місце на ринку.

Вона почала створювати виключно AR для брендів – і продовжує нарощувати сильну внутрішню творчу команду (включаючи 3D-митців), але розширила свою діяльність, щоб запропонувати програмне забезпечення переважно для малого бізнесу для використання та створення власного бренду AR. Zappar виявився популярним серед індустрії розваг, наприклад, у розробниках кінореклами та ігор, оскільки він вітає простий інтерактивний досвід. Найкращі додатки AR забезпечать захоплюючу гру, пов'язану з іншими онлайн-досвідами та обмін фотографіями, відео та інформацією.

3D-художники, такі як розробники ігор і графічні дизайнери, можуть створювати 3D-моделі на добре знайомих їм програмних платформах (наприклад, Unreal Engine і Maya), а потім передавати файли в Zapworks Studio, щоб створити досвід доповненої реальності.

- Aurasma доступна кожному, хто має адресу електронної пошти. Все, що потрібно зробити, це створити обліковий запис, створювати вміст за допомогою веб-студії перетягування (яка також вимірює комерційні результати), поділитися ним із клієнтами або шанувальниками та завантажити програму. Ви навіть можете створити власний екземпляр програми Aurasma з білою міткою або вбудувати цю технологію в нові чи існуючі програми. Такі бренди, як Disney і AMC Theaters, уже використовують його.

Aurasma орієнтована на маркетологів, агентств, розробників, викладачів та студентів. Перегляньте відео нижче, щоб дізнатися, як створити AR в Aurasma Studio.

					ІАЛЦ.467100.003 ПЗ	Арк.
						34
Зм.	Арк.	№ докум.	Підпис	Дата		

2.2.2 Програмне накладання графічного контенту для створення AR

По факту AR це накладання графічного контенту та відео, тому з реалізацією AR шляхом створення коду може впоратись кожна мова програмування, що має можливість виконувати маніпуляції з фото або відео.

Одна з мов, що має широкий потенціал роботи з графічними об'єктами є Python. Python — це високорівнева, інтерпретована мова програмування загального призначення. Його філософія дизайну підкреслює читабельність коду з використанням значних відступів.

Python динамічно типується і збирає сміття. Він підтримує декілька парадигм програмування, включаючи структуроване (зокрема процедурне), об'єктно-орієнтоване та функціональне програмування. Його часто описують як мову з «батареями» через його повну стандартну бібліотеку.

Гвідо ван Россум почав працювати над Python наприкінці 1980-х як наступник мови програмування ABC і вперше випустив його в 1991 році як Python 0.9.0. Python 2.0 був випущений у 2000 році і представив нові функції, такі як осмислення списків, збирання сміття з визначенням циклу, підрахунок посилань та підтримка Unicode. Python 3.0, випущений у 2008 році, був великою версією, яка не повністю сумісна з попередніми версіями. Python 2 було припинено з версією 2.7.18 у 2020 році.

Python постійно займає місце як одна з найпопулярніших мов програмування.

Велика стандартна бібліотека Python надає інструменти, що підходять для багатьох завдань, і зазвичай називають її однією з найбільш сильних сторін. Для інтернет-додатків підтримується багато стандартних форматів і протоколів, таких як MIME і HTTP. Він включає модулі для створення графічних інтерфейсів користувача, підключення до реляційних баз даних, генерування псевдовипадкових чисел, арифметики з десятковими знаками довільної точності, маніпулювання регулярними виразами та модульного тестування.

					ІАЛЦ.467100.003 ПЗ	Арк.
						35
Зм.	Арк.	№ докум.	Підпис	Дата		

Деякі частини стандартної бібліотеки охоплюються специфікаціями — наприклад, реалізація інтерфейсу шлюзу веб-сервера (WSGI) `wsgiref` слідує PEP 333 — але більшість визначається їхнім кодом, внутрішньою документацією та наборами тестів. Однак, оскільки більшість стандартної бібліотеки є кросплатформним кодом Python, лише деякі модулі потребують змін або переписування для варіантів реалізацій.

Крім того Python має багато доповняльних бібліотек з математичною логікою, з інструментами для метематичного корегування та редагування отриманого зображення та роботою напряду з пристроями вводу наприклад камерою, клавіатурою, мікрофоном та іншими.

Основним пакетом для наукових обчислень за допомогою Python є NumPy. NumPy — це бібліотека Python, яка використовується для роботи з масивами. Він також має функції для роботи в області лінійної алгебри, перетворення Фур'є та матриць. NumPy був створений у 2005 році Тревісом Оліфантом. Це проект з відкритим кодом, і тому його можна використовувати вільно.

У Python у нас є списки, які служать меті масивів, але вони повільно обробляються. NumPy має на меті надати об'єкт масиву до 50 разів швидше, ніж традиційні списки Python. Об'єкт масиву в NumPy називається `ndarray`, він надає безліч допоміжних функцій, які роблять роботу з `ndarray` дуже легкою. Ці масиви являють собою стрибкоподібні уявлення про пам'ять. На відміну від вбудованої в Python структури даних списку, ці масиви однотипні: усі елементи одного масиву мають бути одного типу.

Такі масиви також можуть бути переглядами буферів пам'яті, виділених розширеннями C/C++, Cython і Fortran для інтерпретатора CPython без необхідності копіювати дані, що забезпечує певну сумісність із існуючими числовими бібліотеками. Ця функціональність використовується пакетом SciPy, який містить ряд таких бібліотек (зокрема, BLAS і LAPACK). NumPy має вбудовану підтримку відображених у пам'яті `ndarrays`. Оскільки зображення це по факту масив чисел то використання NumPy дуже

					ІАЛЦ.467100.003 ПЗ	Арк.
						36
Зм.	Арк.	№ докум.	Підпис	Дата		

прискорює роботу з математичною частиною редагування та конкатинації зображень.

У Python також є бібліотека для роботи з зображеннями з пристроїв вводу таких як камера. Ця бібліотека називається OpenCV. OpenCV (Open Source Computer Vision Library) — бібліотека комп'ютерного бачення та машинного навчання з відкритим вихідним кодом. OpenCV створено для забезпечення загальної інфраструктури для додатків комп'ютерного зору та для прискорення використання машинного сприйняття в комерційних продуктах. Будучи ліцензованим продуктом BSD, OpenCV дозволяє легко використовувати та змінювати код[2].

Бібліотека має понад 2500 оптимізованих алгоритмів, які включають повний набір як класичних, так і найсучасніших алгоритмів комп'ютерного зору та машинного навчання. Ці алгоритми можна використовувати для виявлення та розпізнавання облич, ідентифікації об'єктів, класифікації дій людини на відео, відстеження рухів камери, відстеження рухомих об'єктів, вилучення 3D-моделей об'єктів, створення тривимірних хмар точок із стереокамер, з'єднання зображень разом для отримання високої роздільної здатності. зображення всієї сцени, знаходити схожі зображення з бази даних зображень, видаляти червоні очі із зображень, зроблених за допомогою спалаху, слідкувати за рухами очей, розпізнавати пейзаж і встановлювати маркери для накладання на них доповненої реальності тощо. OpenCV має понад 47 тисяч користувачів спільноти, а кількість завантажень перевищує 18 мільйонів. Бібліотека широко використовується компаніями, науковими групами та державними органами[6].

Поряд із такими відомими компаніями, як Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota, які використовують бібліотеку, є багато стартапів, таких як Applied Minds, VideoSurf і Zeitera, які широко використовують OpenCV. Застосування OpenCV охоплюють широкий діапазон від зшивання зображень у режимі перегляду вулиць, виявлення вторгнень у

					ІАЛЦ.467100.003 ПЗ	Арк.
						37
Зм.	Арк.	№ докум.	Підпис	Дата		

відеоспостереження в Ізраїлі, моніторингу шахтного обладнання в Китаї, допомоги роботам орієнтуватися та підбирати об'єкти в Willow Garage, виявлення нещасних випадків, пов'язаних із потопленням у басейні в Європі, запуску інтерактивного мистецтва в Іспанія та Нью-Йорк перевіряють злітно-посадкові смуги на предмет сміття в Туреччині, перевіряють етикетки на продуктах на фабриках по всьому світу для швидкого виявлення обличчя в Японії.

Він має інтерфейси C++, Python, Java і MATLAB і підтримує Windows, Linux, Android і Mac OS. OpenCV схиляється здебільшого до програм бачення в реальному часі та використовує переваги інструкцій MMX та SSE, якщо вони доступні. Зараз активно розробляються повнофункціональні інтерфейси CUDA та OpenCL. Існує понад 500 алгоритмів і приблизно в 10 разів більше функцій, які створюють або підтримують ці алгоритми. OpenCV написаний на C++ і має шаблонний інтерфейс, який безперебійно працює з контейнерами STL.[1]

2.3 Матричні коди

Крім усього сказаного для реалізації AR необхідно тако ж розібратися з закодованим матричним кодом у графічному об'єкті. Це потрібно для передавання інформації через зображення. Одним з таких графічних об'єктів є QR код.

Матричний код — це двовимірний штрих-код, який можна надрукувати у вигляді квадрата або прямокутний символ, що складається з окремих точок або квадратів. Це представлення є впорядкованою сіткою темних і світлих точок, облямованих візерунком шукача. Шаблон шукача частково використовується для визначення орієнтації та структура символу. Дані кодуються за допомогою серії темних або світлих точок на основі заздалегідь визначеного розміру. Розмір цих точок відомий як X-вимір.

					ІАЛЦ.467100.003 ПЗ	Арк.
						38
Зм.	Арк.	№ докум.	Підпис	Дата		

Матричний код складається з двох окремих частин: шаблон пошуку, який використовується сканером для визначення місцезнаходження символу та самих закодованих даних. Шаблон шукача визначає форму (квадрат або прямокутник), розмір, X-вимір і число рядків і стовпців у символі. Він має функцію, подібну до допоміжних шаблонів (Start, Stop і центральні візерунки) у штрих-кодів EAN/UPC і дозволяє сканеру ідентифікувати символ як GS1 DataMatrix. Суцільний темний колір називається «образом шукача L». Його в першу чергу використовують для визначення розміру, орієнтації і спотворення символу. Дві інші сторони візерунка шукача - це чергування світлих і темних елементів, відомих як «Годинник». Це визначає основну структуру символу, а також може допомогти визначити його розмір і спотворення. Дані потім кодуються в матриці в рамках шаблону Finder[5].

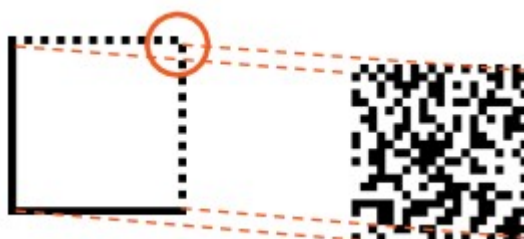


Рис. 2.1 Шаблон Finder і дані

2.3.1 Матричний код QR

QR-код можна зустріти повсюду – від сайтів до рекламних щитів. Вони нагадують штрих-коди, тільки замість вертикальних смуг складаються з маленьких квадратиків. Вони є все частіше і стають все більш популярними, тому багато смартфонів тепер випускаються з вбудованим сканером QR-кодом.

QR – це скорочення від Quick Response, що означає «швидкий відклик».

Незважаючи на свій розмір QR-код, здатний вмістити велику кількість даних. Но не це головне. Скануючи QR-код, ви отримуєте доступ до цього одразу – звідси і назва.

Звичайний штрих-код читається тільки в одному напрямленні – слева направо. Тому він може зберігати порівняно невеликий обсяг інформації – як правило, це букви та цифри. QR-код читається в двох напрямках (поєтому його також називають «двумерним штрих-кодом») – по горизонталі і по вертикалі. Це дозволяє зберігати в нем більше даних. Целью розробників було створено знавчий, сканований під будь-яким кутом формату, щоб код було легко помітити і не потрібно було тратити час, вибираючи правильне положення сканера. Так вони прийшли до всього відомого квадратного коду, який використовується на цей день.

Розробка першого QR-кода заняла більше року. Він міг вмістити до 7000 цифр і ієрогліфів кандзи і рахувався в десять разів швидше, чим звичайний штрих-код[7].

У зорі QR-коду зашифрована послідовність даних у двійковому форматі.

Зчитуючий пристрій розпізнає стандартний QR-код за трьома квадратними мітками, розташованими по його кутах. Виявивши їх, сканер зчитує все, що знаходиться у визначеному ними квадраті, а потім аналізує QR-код, представляючи його у вигляді сітки.

Кожному окремому осередку сітки надається значення залежно від кольору (чорний або білий). Потім осередки групуються у більші візерунки.

2.3.2 Код-мітка ArUco

ARTag — це довідкова система маркерів для підтримки доповненої реальності. Їх можна використовувати для полегшення появи віртуальних об'єктів, ігор та анімацій у реальному світі. Як і попередня система ARToolKit, вони надають можливості відстеження відео, які обчислюють положення та орієнтацію камери щодо фізичних маркерів у реальному часі. Як тільки положення камери відоме, віртуальну камеру можна розташувати в тій же точці, виявляючи віртуальний об'єкт у місці розташування ARTag. Таким

					ІАЛЦ.467100.003 ПЗ	Арк.
						40
Зм.	Арк.	№ докум.	Підпис	Дата		

чином, він вирішує дві ключові проблеми доповненої реальності: відстеження точки зору та взаємодія віртуальних об'єктів.

У Науковій лабораторії Марса з'являється позначка ART. Подібна техніка використовується в додатку NASA для смартфонів Spacecraft 3D як освітній інструмент.

Одним з таких є Бібліотека ARUCO була розроблена групою Ava з Університету Кордови (Іспанія). Він забезпечує оцінку 3D пози в режимі реального часу на основі маркерів за допомогою маркерів AR.

					ІАЛЦ.467100.003 ПЗ	Арк.
						41
Зм.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВОК ДО РОЗДІЛУ 2

Було розглянуто напрямки використання, можливості та способи створення AR.

В останні роки технологія доповненої реальності привертає велику увагу дослідників. Поєднана за допомогою комп'ютерного зору та технології штучного інтелекту технологія доповненої реальності показала сильний імпульс розвитку. Точність реєстрації відстеження та обладнання відображення продуктивність і характер взаємодії людини з комп'ютером були значно покращені. Однак можна помітити, що в технології доповненої реальності ще потрібно вирішити багато проблем. З точки зору Технологія реєстрації відстеження, потоковий метод реєстрації відстеження може використовувати лише невелику кількість інформації, наприклад, інформація про особливості, що призводить до неповноти розуміння системи до навколишнього середовища. З точки зору технології відображення, розмірів і ціни - окуляри доповненої реальності, які можуть забезпечити користувачам високе відчуття занурення, не можуть задовільнити запит громадськості. Що стосується режиму взаємодії, то більш природна і багатокористувацька доповнена реальність технологія взаємодії ще не досліджена.

Після огляду можливостей та напрямків використанні точно зрозуміло, що така технологія користується популярністю великих компаній та має величезний потенціал у майбутньому.

Розглянуті способи реалізації технології AR було вирішено для написання продукту використовувати мову програмування Python через широкий спектр вибору потрібних бібліотек та можливостей мови. Було знайдено бібліотеки що ідеально підходять для створення зазначеного продукту, а саме, програми, що конкатенує зображення з пристрою вводу - камери та потокового зображення у один графічний елемент. Оскільки OpenCV дозволяє напряду виконувати обрахунки для роботи з графічними

					ІАЛЦ.467100.003 ПЗ	Арк.
						42
Зм.	Арк.	№ докум.	Підпис	Дата		

зображеннями то було вирішено використовувати саме цю бібліотеку та маркери ArUco

					ІАЛЦ.467100.003 ПЗ	Арк.
						43
Зм.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ДЕТАЛІ РОЗРОБКИ ПРОГРАМИ

Відповідно до досліджених матеріалів та обраного способу реалізації ми можемо перейти до фази розробки програмного продукту. Продукт складається з частини потокової передачі даних та з графічної частини конкатинації потокового зображення з зображенням з камери. Тому спочатку створимо поточкову трансляцію зображення а після конкатинуємо її з зображенням з камери.

3.1 Розробка передачі живого потокового зображення

Далі буде описано основні класи та алгоритм передачі зображення.

3.1.1 Клас StreamingClient

StreamingClient це абстрактний клас для загального потокового клієнта.

У ньому я створив такі атрибути:

`_host` - який приймає значення `str` та є адресою хоста для підключення

`_port` - який приймає значення `int` та є портом для підключення

`_running` - який приймає значення `bool` та є індикатором того чи відбувається потокова трансляція

`_encoding_parameters` - це список параметрів кодування для бібліотеки OpenCV

`_client_socket` - за значення якого сприймається основний клієнтський сокет

```
self.__host = host
self.__port = port
self.__configure()
self.__running = False
self.__client_socket = socket.socket(
    socket.AF_INET, socket.SOCK_STREAM)
```

Рис. 3.1 Атрибути класу StreamingClient

Крім того було створено такі методи:

`_client_streaming()` - основний метод потокової передачі даних клієнта

```
def _client_streaming(self):|
    self.__client_socket.connect((self.__host, self.__port))
    while self.__running:
        frame = self._get_frame()
        result, frame = cv2.imencode(
            '.jpg', frame, self.__encoding_parameters)
        data = pickle.dumps(frame, 0)
        size = len(data)

        try:
            self.__client_socket.sendall(struct.pack('>L', size) + data)
        except ConnectionResetError:
            self.__running = False
        except ConnectionAbortedError:
            self.__running = False
        except BrokenPipeError:
            self.__running = False

    self._cleanup()
```

Рис. 3.2 Метод `_client_streaming()`

У ньому ми можемо побачити як потік під'єднується до хоста за адресою для підключення та портом.

Після чого отримується окремий фрейм(зображення), який далі кодується у бінарному коді а далі іде перевірка чи під'єднання до хоста все ще дійсне

`_configure()` - метод, що встановлює основні конфігурації (перевизначені дочірніми класами)

```
def _configure(self):|
    self.__encoding_parameters = [int(cv2.IMWRITE_JPEG_QUALITY), 90]
```

Рис. 3.3 Метод `_configure()`

Зараз у ньому записана функція що визначає параметри кодування за дефолтом.

`_get_frame()` - повертає кадр, який буде надіслано на сервер (замінено дочірніми класами)

За основу повертає None.

```
def _get_frame(self):  
    return None
```

Рис. 3.4 Метод _get_frame()

_cleanup() - метод, що очищає всі ресурси і закриває все.

```
def _cleanup(self):  
    cv2.destroyAllWindows()
```

Рис. 3.5 Метод _cleanup()

start_stream() -запускає клієнтську потокову передачу у новому потоці

Спочатку метод перевіряє чи вже запущена трансляція з цим клієнтом, і якщо ні, то ставить флажок на атрибуті _running та починає трансляцію

```
def start_stream(self):  
  
    if self.__running:  
        print("Client is already streaming!")  
    else:  
        self.__running = True  
        client_thread = threading.Thread(target=self.__client_streaming)  
        client_thread.start()
```

Рис. 3.5 Метод start_stream()

stop_stream() - зупиняє трансляцію клієнта перевіркою та зміною атрибута _running

```
def stop_stream(self):  
    if self.__running:  
        self.__running = False  
    else:  
        print("Client not streaming!")
```

Рис. 3.5 Метод stop_stream()

3.1.2 Клас ScreenShareClient

Це клас для передачі потової передачі зображення екрану. Він наслідує методи абстрактного класу StreamingClient та переписує їх.

У ньому присутні такі атрибути:

`_host` - адреса хоста для підключення

`_port` - порт для підключення

`_running` - вказує, чи клієнт уже транслюється чи ні

`_encoding_parameters` - список параметрів кодування для OpenCV

`_client_socket` - основний клієнтський сокет

`_x_res` - роздільна здатність по горизонталі

`_y_res` - роздільна здатність по вертикалі

```
def __init__(self, host, port, x_res=1024, y_res=576):
    self.__x_res = x_res
    self.__y_res = y_res
    super(ScreenShareClient, self).__init__(host, port)
```

Рис. 3.6 Атрибути класу ScreenShareClient

Наступні методи переписують методи абстрактного класу:

`_get_frame()` - повертає кадр знімку екрану, який потрібно надіслати на сервер

У ньому за допомогою бібліотеки `pyautogui` програма отримує скріншот екрану, перекодує його у `np.array` та закодує його для роботи із бібліотекою `cv2`

```
def _get_frame(self):
    screen = pyautogui.screenshot()
    frame = np.array(screen)
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    frame = cv2.resize(frame, (self.__x_res, self.__y_res),
                       interpolation=cv2.INTER_AREA)
    return frame
```

Рис. 3.7 Метод `_get_frame()`

Всі інші унаслідковані класи залишаються незмінними.

3.1.3 Клас StreamingServer

Це клас, що приймає та відображає трансляцію отриману з клієнту. У ньому створені наступні атрибути:

`_host` - приймає значення `str` та є адресою серверу

`_port` - приймає значення `int` та є портом серверу

`_quick_key` - приймає значення `ch`, що є клавішею, яку потрібно натиснути, щоб закрити з'єднання

`_running` - показує, запущено сервер чи ні з типом `bool`

`_block` - базове блокування, яке використовується для синхронізації потоків

`_server_socket` - головний серверний сокет

```
def __init__(self, host, port, slots=8, quit_key='q'):
    self.__host = host
    self.__port = port
    self.__slots = slots
    self.__used_slots = 0
    self.__running = False
    self.__quit_key = quit_key
    self.__block = threading.Lock()
    self.__server_socket = socket.socket(
        socket.AF_INET, socket.SOCK_STREAM)
    self.__init_socket()
```

Рис. 3.8 Атрибути класу StreamingServer

У цьому класі були написані наступні методи:

`_init_socket()` - метод, який прив'язує сокет сервера до хосту та порту

```
def __init_socket(self):
    self.__server_socket.bind((self.__host, self.__port))
```

Рис. 3.9 Метод `_init_socket()`

`_server_listening()` - метод, який чекає на нові з'єднання та починає потік якщо є вільні слоти та з'єднання прийнято.

```
def _server_listening(self):
    self.__server_socket.listen()
    while self.__running:
        self.__block.acquire()
        connection, address = self.__server_socket.accept()
        if self.__used_slots >= self.__slots:
            print("Connection refused! No free slots!")
            connection.close()
            self.__block.release()
            continue
        else:
            self.__used_slots += 1
            self.__block.release()
            print(connection)
            thread = threading.Thread(
                target=self.__client_connection, args=(connection, address,))
            thread.start()
```

Рис. 3.10 Метод `_server_listening()`

`_client_connctcion()` - Основний метод обробки клієнтських потокових зображень. Обробляє окремі клієнтські підключення та обробляє їх потокові дані. Саме у цьому методі пізніше буде реалізовано AR технологію та конкатинацію зображень. Зраз він просто розкодує та розпаковує отримане зображення від клієнту

```
def _client_connection(self, connection, address):
    payload_size = struct.calcsize('>L')
    data = b''
    cap = cv2.VideoCapture(0)
    while True:
        break_loop = False

        while len(data) < payload_size:
            received = connection.recv(4096)
            if received == b'':
                connection.close()
                self.__used_slots -= 1
                break_loop = True
                break
            data += received

        if break_loop:
            break

        packed_msg_size = data[:payload_size]
        data = data[payload_size:]

        msg_size = struct.unpack(">L", packed_msg_size)[0]

        while len(data) < msg_size:
            data += connection.recv(4096)

        frame_data = data[:msg_size]
        data = data[msg_size:]

        frame = pickle.loads(
            frame_data, fix_imports=True, encoding="bytes")
        img1 = cv2.imdecode(frame, cv2.IMREAD_COLOR)
        succ, img = cap.read()

        if cv2.waitKey(1) == ord(self.__quit_key):
            connection.close()
            self.__used_slots -= 1
            break
```

Рис. 3.11 Метод `_client_connctcion()`

start_server() - запускає сервер у новому потоці та запускає сервер, якщо він ще не запущений.

```
def start_server(self):  
    if self.__running:  
        print("Server is already running")  
    else:  
        self.__running = True  
        server_thread = threading.Thread(target=self.__server_listening)  
        server_thread.start()
```

Рис. 3.12 Метод start_server()

stop_server() - зупиняє сервер і закриває всі з'єднання

```
def stop_server(self):  
    if self.__running:  
        self.__running = False  
        closing_connection = socket.socket(  
            socket.AF_INET, socket.SOCK_STREAM)  
        closing_connection.connect((self.__host, self.__port))  
        closing_connection.close()  
        self.__block.acquire()  
        self.__server_socket.close()  
        self.__block.release()  
    else:  
        print("Server not running!")
```

Рис. 3.13 Метод stop_server()

Тож вже написані класи та методи, що отримують зображення, кодують його, запаковують, відправляють на сервер, де його отримують, розпаковують та відображають. Залишається прописати логіку конкатинації зображень для AR.

3.2 Створення AR способом конкатинації зображень

Перед створенням логіки конкатинації зображень, необхідно отримати зображення з камери для цього використаємо бібліотеку OpenCV та запишемо отримане зображення в змінну cap

```
cap = cv2.VideoCapture(0)
```

Рис. 3.13 Отримання зображення з камери

					ІАЛЦ.467100.003 ПЗ	Арк.
						50
Зм.	Арк.	№ докум.	Підпис	Дата		

3.2.1 Робота з маркерами ArUco

Після того як ми отримали зображення з камери, необхідно визначити місце де необхідно вставити зображення нашої трансляції. Це не можуть бути сталі координати атже камера посто змінює картинку тож це буде вже не доповнена реальність а просто вставлена кртинка, тому ми викоритсаємо ArUco маркери для визначення 4 точок на зображенні камери на місце яких пізніше вставимо зображення трансляції. Для створення таких маркерів скористаємося сервісом chev.me та закодуємо у 1 маркері цифру 0, у 2гому цифру 1, у третьому цифру 2 а у 4тому цифру 3 для того, щоб роузміти їх порядок.

Dictionary: 6x6 (50, 100, 250, 1000) ▼
Marker ID: 0
Marker size, mm: 100

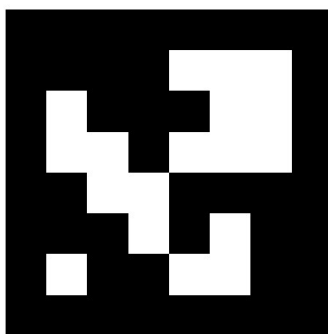


Рис. 3.14 Маркер Aruco #1

Dictionary: 6x6 (50, 100, 250, 1000) ▼
Marker ID: 1
Marker size, mm: 100

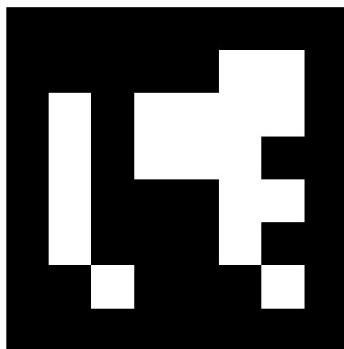


Рис. 3.15 Маркер Aruco #2

Dictionary: 6x6 (50, 100, 250, 1000) ▾
 Marker ID: 2 ▾
 Marker size, mm: 100



Рис. 3.16 Маркер Aruco #3

Dictionary: 6x6 (50, 100, 250, 1000) ▾
 Marker ID: 3 ▾
 Marker size, mm: 100

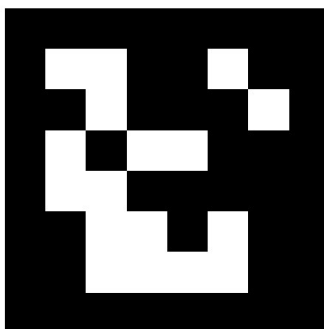


Рис. 3.17 Маркер Aruco #4

Для знаходження маркерів на зображенні, використаємо вбудовану бібліотеку у OpenCV - aruco.

Бібліотека OpenCV може виявити 21 різний тип маркерів ArUco.

Задамо параметри маркерів, та знайдемо їх координати на зображенні, після чого запишемо координати у змінну `res_boxes`. А саме вбудовану бібліотек маркерів `DICT_6x6_250` замість `DICT_4x4_50` за замовчуванням.

Це є унікальний ідентифікатор змінної, присвоєний кожному типу словника маркера.

```

frame = pickle.loads(
    frame_data, fix_imports=True, encoding="bytes")
img1 = cv2.imdecode(frame, cv2.IMREAD_COLOR)
succ, img = cap.read()

markerSize = 6
totalMarkers = 250
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
key = getattr(aruco, 'DICT_6X6_250')
aruco_dict = aruco.Dictionary_get(key)
aruco_param = aruco.DetectorParameters_create()

res_boxes, res_ids, rej = aruco.detectMarkers(
    img_gray, aruco_dict, parameters=aruco_param)
my_polygon = [0, 0, 0, 0]

```

Рис. 3.18 Детекція маркерів на зображенні

3.2.2 Алгоритм накладання зображень

Далі необхідно зробити співвідношення між координатами маркерів та координатами трансльованого зображення `my_polygon`. Через те, що маркери у кодованому зображенні можуть відображатися не в такій послідовності як потрібно, то необхідно також провести їх сортування. Після чого ми дізнаємося висоту та широту зображення

```

if res_boxes:
    polygon_ids = [[i[0][0], k[0]]
                   for i, k in zip(res_boxes, res_ids)]
    print(polygon_ids)
    if len(polygon_ids) == 4:
        for i in polygon_ids:
            if i[1] == 0:
                my_polygon[3] = i[0]
            elif i[1] == 1:
                my_polygon[2] = i[0]
            elif i[1] == 2:
                my_polygon[0] = i[0]
            elif i[1] == 3:
                my_polygon[1] = i[0]
        list1 = []
        list2 = []
        count = 0
        for i in my_polygon:
            if count == 3 and i[0] < my_polygon[1][0]:
                list1.insert(0, i)
            else:
                list1.append(i)
            list1.append(i)
            if count == 3:
                if i[0] < my_polygon[1][0]:
                    list2.insert(0, i)
                    list2[2], list2[3] = list2[3], list2[2]
                else:
                    list2.insert(2, i)
            else:
                list2.append(i)
            count += 1
        h1, w1 = img1.shape[:2]

```

Рис. 3.19 Сортування отриманих координат

Тепер у нас є координати обох зображень. Ми обчислимо матрицю гомографії за допомогою функції `cv2.findHomography()`.

Гомографія пов'язує перетворення між двома площинами (з точною мірою масштабного коефіцієнта)[1]:

$$s \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Рис. 3.20 Гомографічна матриця

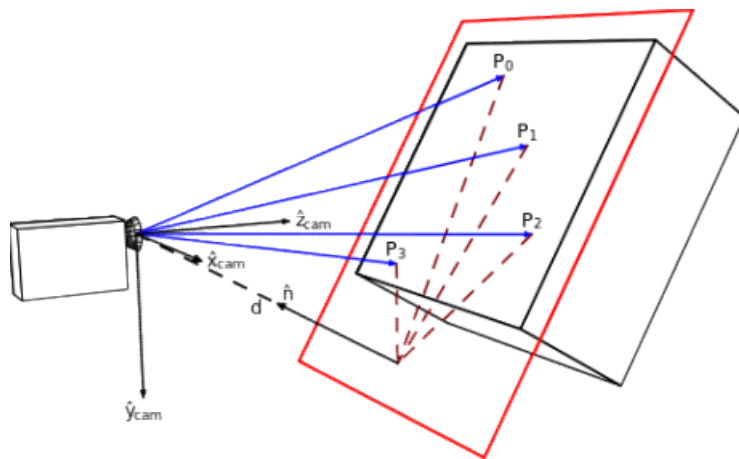


Рис. 3.21 Візуалізація гомографії

Тож використаємо гомографічну матрицю для знаходження перспективи зображення.

Pts1 - це матриця координат зображення з камери.

Pts2 - це матриця координат зчитана з маркерів тобто координат трансльованого зображення.

```
pts1 = np.float32([[0, 0], [0, h1], [w1, 0], [w1, h1]])
pts2 = np.float32(list2)

h, mask = cv2.findHomography(pts1, pts2, cv2.RANSAC, 5.0)
```

Рис. 3.22 Знаходження гомографічної матриці

Тепер ми скористаємося функцією `cv2.warpPerspective()`, яка візьме вхідні дані з камери(`img1`), матрицю гомографії(`h`), ширину та висоту трансльованого зображення, і в результаті отримаємо трансльоване зображення у формі та перспективі маркерів.

```
height, width, channels = img.shape
im1Reg = cv2.warpPerspective(img1, h, (width, height))
```

Рис. 3.23 Трансформування зображення

Зробимо маску за допомогою `np.zeros()` розміру трансльованого зображення.

Тепер ми заповнюємо маску білою зоною за допомогою функції `cv2.fillConvexPoly()`, яка поверне трансформований білий прямокутник у розмір з трансформованого трансльованого зображення

Після трансформації зображення ми використаємо `cv2.bitwise_or()`, щоб отримати остаточний результат.

```
mask2 = np.zeros(img.shape, dtype=np.uint8)

roi_corners2 = np.int32(list1)

channel_count2 = img.shape[2]
ignore_mask_color2 = (255,) * channel_count2

cv2.fillConvexPoly(mask2, roi_corners2, ignore_mask_color2)

mask2 = cv2.bitwise_not(mask2)
masked_image2 = cv2.bitwise_and(img, mask2)
# Using Bitwise or to merge the two images
img = cv2.bitwise_or(im1Reg, masked_image2)
```

Рис. 3.24 Конкатинація зображень

ВИСНОВОК ДО РОЗДІЛУ 3

В результаті проведеної роботи щодо розробки потокового передавання екрану та відображення його у доповненій реальності було реалізовано потокове передавання зображення екрану, кодування та розкодування зображення, перетворення зображення у перспективі та використання гомографічної матриці.

Для початку було виконаний алгоритм трансляції потокового зображення. У якому програма спочатку робить знімок екрану, шифрує його, після чого відправляє його на дивайс отримувача. Де програма на дивайсі отримувача, отримує пакет з зашифрованим зображенням, розпаковує та розкодує його після чого реалізує алгоритм накладення зображення та відображає його.

Після алгоритму трансляції зображення було створено алгоритм візуалізації доповненої реальності.

Після налагодження зв'язку з сервером, програма отримує окремо кожен фрейм з камери, після чого вираховує координати маркерів на отриманому зображенні для того щоб координати маркерів стали новими координатами зображення отриманого з трансляції серверу.

Але цього не достатньо, необхідно також враховувати перспективу маркерів, тому було використано гомографічну матрицю для трансформування трансльованого зображення після чого були виконані необхідні графічні дії з перетворенням кольорів для правильної конкатації зображень.

З'єднання між сервером та клієнтом було реалізовано за допомогою передачі по HTTP IP клієнта до сервера.

У результаті вийшло запрограмувати систему передачі потокового зображення комп'ютера та вивести її у режимі доповненої реальності.

					ІАЛЦ.467100.003 ПЗ	Арк.
						56
Зм.	Арк.	№ докум.	Підпис	Дата		

Тож були виконані усі поставленні задачі що до потокової передачі зображення та відображення його у доповненій реальності

					ІАЛЦ.467100.003 ПЗ	Арк.
						57
Зм.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

Під час виконання дипломної роботи було створено систему для потокової передачі зображення екрану комп'ютера. Данна система дає можливість користувачам дистанційного доступу до екрану комп'ютера та відображення його у доповненій реальності.

У першому розділі були розглянуті методи передачі та способи використання трансльованого зображення методом потокової передачі та способи її реалізації. Визначені переваги та недостатки потокової передачі та певних протоколів.

У другому розділі були описані детально алгоритми створення та застосунку доповненої реальності та способи передачі інформації графічно.

У третьому розділі було реалізовані раніше обґрунтовані способи реалізації трансляції зображення та відтворення її у режимі доповненої реальності.

Слід також зазначити, що під час розробки застосунку були враховані всі зазначені на початку вимоги, яким він мав відповідати.

					ІАЛЦ.467100.003 ПЗ	Арк.
						58
Зм.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Гомографічна матриця (Homography matrix) [Електронний ресурс] – Режим доступу до ресурсу:
https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html
2. OpenCV [Електронний ресурс] – Режим доступу до ресурсу:
<https://pypi.org/project/opencv-python/>
3. Client–server model [Електронний ресурс] –
https://en.wikipedia.org/wiki/Client%E2%80%93server_model
4. Yunqiang Chen, (2019). «An overview of augmented reality technology».
5. DataMatrix Guideline (2018) Overview and technical introduction to the use of GS1 DataMatrix//The Global Language of Business.
6. Mehrabiana A.R., Lucase C. A novel numerical optimization algorithm inspired from weed colonization // Ecological informatics. 2006. No 1. P. 355—366.
7. Henryk Josiński, Daniel Kostrzewa, Agnieszka Michalczuk, Adam Świtoński, "The Expanded Invasive Weed Optimization Metaheuristic for Solving Continuous and Discrete Optimization Problems", The Scientific World Journal, vol. 2014, Article ID 831691, 14 pages, 2014.
8. Yang X.-S., Deb S. Cuckoo search via Levy flights // Proc. of the world Congress on Nature & Biologically Inspired Computing (NaBIC 2009), December 2009, India. IEEE Publications, USA, pp. 210—214.
9. Oliver C. Ibe. Levy Processes / Oliver C. Ibe // Markov Processes for Stochastic Modeling (Second Edition) / Oliver C. Ibe., 2013. – С. 329–347.
10. Tuba M., Subotic M., Stanarevic N. Modified cuckoo search algorithm for unconstrained optimization problems // Proc. of the 5th European Computing Conference (ECC'11), Paris, France, April 28—30, 2011. pp. 263—268.

11. Yang X.-S. A New Metaheuristic Bat-Inspired Algorithm, in: Nature Inspired Cooperative Strategies for Optimization (NISCO 2010). Studies in Computational Intelligence. Berlin: Springer, 2010. Vol. 284. P.65-74.
12. Doppler effect [Електронний ресурс] // The Editors of Encyclopaedia Britannica. – 2019. – Режим доступу до ресурсу: <https://www.britannica.com/science/Doppler-effect>.
13. Mucherino A., Seref O. Monkey Search: A Novel Meta-Heuristic Search for Global Optimization // Data Mining, System Analysis and Optimization in Biomedicine, AIP Conference Proceedings. 2007. P. 162—173.
14. Herbert Robbins. Sutton Monro // A Stochastic Approximation Method, 400 - 407, September, 1951
15. Zhao R., Tang W. Monkey Algorithm for Global Numerical Optimization // Journal of Uncertain Systems. 2008. V. 2. N 3. P. 165—176.
16. Bastos-Filho C.J.A., Lima-Neto F. B., Lins A., Nascimento A., Lima M. Fish school search // Nature-Inspired Algorithms for Optimisation. SCI. Springer. Heidelberg. 2009. Vol. 193. P. 261—277.

ДОДАТОК 1

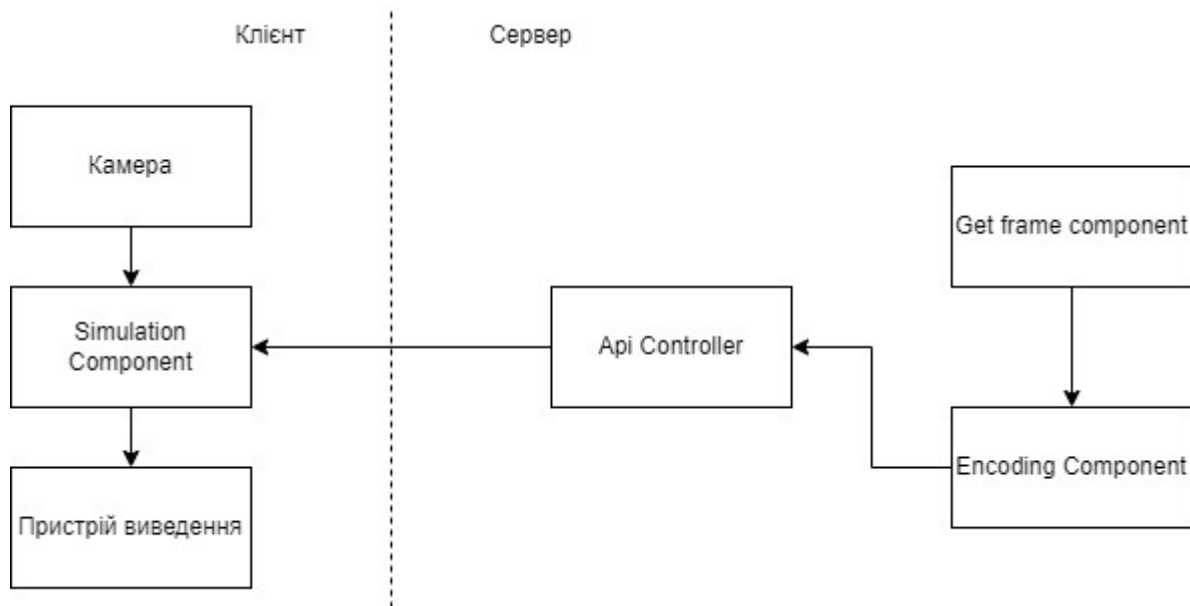
Система дистанційного транслявання зображення комп'ютера

Структурна схема системи

ІАЛЦ.467100.004 Д1

Аркушів 1

Київ 2022 р



					ІАЛЦ.467100.004 Д1								
		№ докум.	Підпис	Дата									
Розробив		Сорожинський М.М.			ЕСистема дистанційного транслявання зображення комп'ютера Структурна схема системи				Літ.	Аркуш	Аркушів		
Перевірив		Калюжний О. О.									1	1	
									НТУУ КПІ ім. Ігоря Сікорського, ФІОТ, ІО-81				
Н. Контр.		Сімоненко В.											
Затвердив													

ДОДАТОК 2

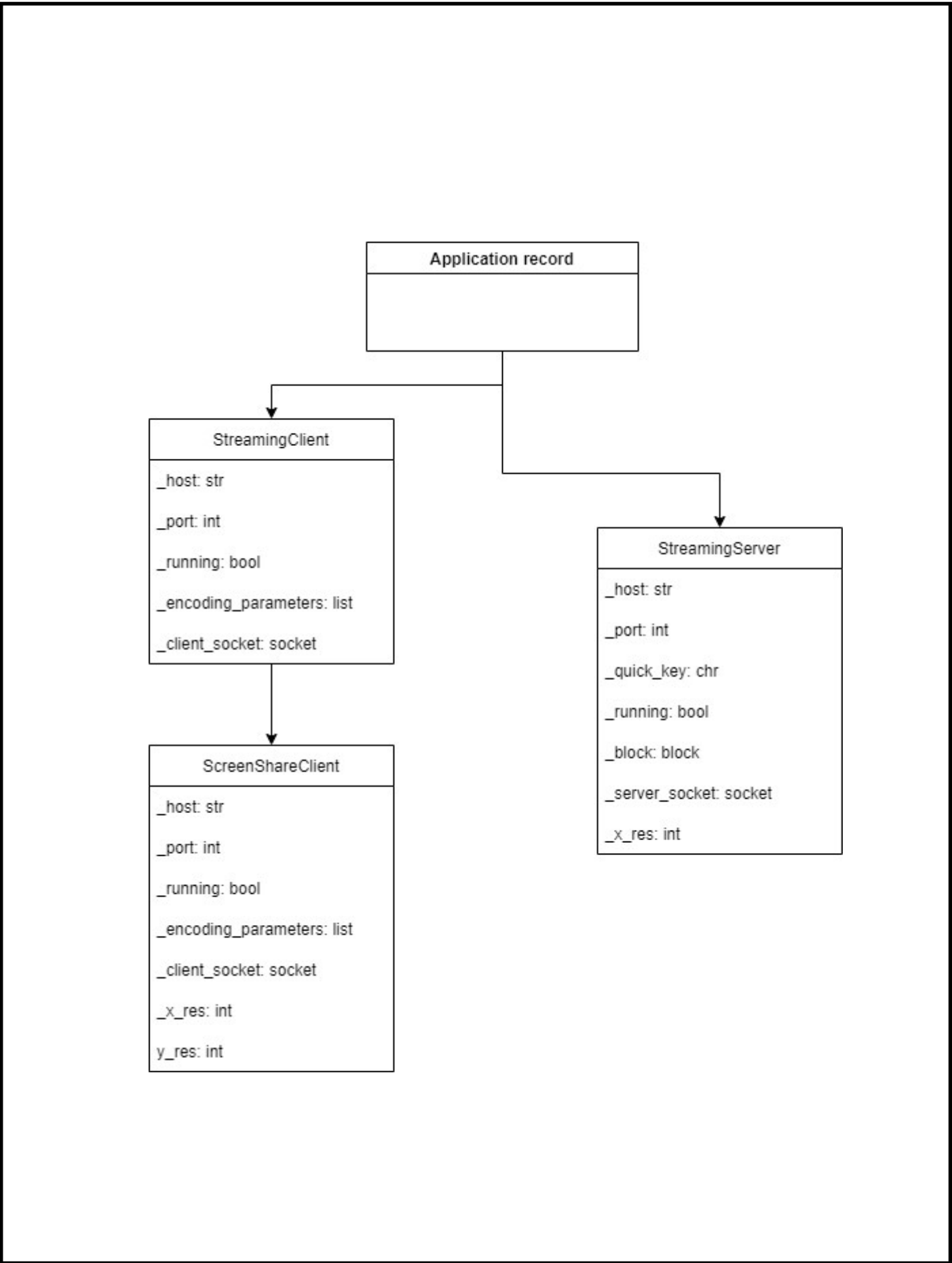
Система дистанційного транслявання зображення комп'ютера

Функціональна схема (діаграма класів)

ІАЛЦ.467100.005 Д2

Аркушів 1

Київ 2022 р



					ІАЛЦ.467100.005 Д2						
		№ докум.	Підпис	Дата							
Розробив	Сорожинський М.М.				Система дистанційного транслявання зображення комп'ютера Функціональна схема (діаграма класів)			Літ.	Аркуш	Аркушів	
Перевірив	Калюжний О.О.									1	1
								НТУУ КПІ ім. Ігоря Сікорського, ФІОТ, ІО-81			
Н. Контр.	Сімоненко В.										
Затвердив											

ДОДАТОК 3

Система дистанційного транслявання зображення комп'ютера

Алгоритм дій програмного забезпечення

ІАЛЦ.467100.006 ДЗ

Аркушів 1

Київ 2022 р



					ІАЛЦ.467100.006 ДЗ						
		№ докум.	Підпис	Дата							
Розробив		Сорожинський М.М.			Система дистанційного транслявання зображення комп'ютера Алгоритм дій програм- ного забезпечення			Літ.	Аркуш	Аркушів	
Перевірив		Калюжний О.О.								1	1
Н. Контр.		Сімоненко В.						НТУУ КПІ ім. Ігоря 66			
Затвердив								Сікорського, ФІОТ, ІО-81			

ДОДАТОК 4

Система дистанційного транслявання зображення комп'ютера

Текст програмного коду

ІАЛЦ.467100.007 Д4

Аркушів 22

Київ 2022 р

```
import cv2
import pyautogui
import numpy as np
import cv2.aruco as aruco
import numpy as np
```

```
import socket
import pickle
import struct
import threading
```

```
class StreamingServer:
```

```
    """
```

```
    Class for the streaming server.
```

```
    Attributes
```

```
    -----
```

```
    Private:
```

```
        __host : str
```

```
            host address of the listening server
```

```
        __port : int
```

```
            port on which the server is listening
```

```
        __slots : int
```

```
            amount of maximum avaialable slots (not ready yet)
```

```
        __used_slots : int
```

```
            amount of used slots (not ready yet)
```

```
        __quit_key : chr
```

					ІАЛЦ.467100.07 Д4	Арк.
						1
Зм.	Арк.	№ докум.	Підпис	Дата		

key that has to be pressed to close connection

__running : bool

indicates if the server is already running or not

__block : Lock

a basic lock used for the synchronization of threads

__server_socket : socket

the main server socket

Methods

Private:

__init_socket : method that binds the server socket to the host and

port

__server_listening: method that listens for new connections

__client_connection : main method for processing the client streams

Public:

start_server : starts the server in a new thread

stop_server : stops the server and closes all connections

"""

TODO: Implement slots functionality

def __init__(self, host, port, slots=8, quit_key='q'):

self.__host = host

self.__port = port

					ІАЛЦ.467100.07 Д4	Арк.
						2
Зм.	Арк.	№ докум.	Підпис	Дата		

```

self.__slots = slots
self.__used_slots = 0
self.__running = False
self.__quit_key = quit_key
self.__block = threading.Lock()
self.__server_socket = socket.socket(
    socket.AF_INET, socket.SOCK_STREAM)
self.__init_socket()

def __init_socket(self):
    self.__server_socket.bind((self.__host, self.__port))

def start_server(self):
    if self.__running:
        print("Server is already running")
    else:
        self.__running = True
        server_thread = threading.Thread(target=self.__server_listening)
        server_thread.start()

def __server_listening(self):
    self.__server_socket.listen()
    while self.__running:
        self.__block.acquire()
        connection, address = self.__server_socket.accept()
        if self.__used_slots >= self.__slots:
            print("Connection refused! No free slots!")
            connection.close()
            self.__block.release()
            continue

```

					ІАЛЦ.467100.07 Д4	Арк.
						3
Зм.	Арк.	№ докум.	Підпис	Дата		

```

        else:
            self.__used_slots += 1
        self.__block.release()
        print(connection)
        thread = threading.Thread(
            target=self.__client_connection, args=(connection,
address,))

        thread.start()

def stop_server(self):
    if self.__running:
        self.__running = False
        closing_connection = socket.socket(
            socket.AF_INET, socket.SOCK_STREAM)
        closing_connection.connect((self.__host, self.__port))
        closing_connection.close()
        self.__block.acquire()
        self.__server_socket.close()
        self.__block.release()
    else:
        print("Server not running!")

def __client_connection(self, connection, address):
    payload_size = struct.calcsize('>L')
    data = b""
    cap = cv2.VideoCapture(0)
    while True:

        break_loop = False

```

					ІАЛЦ.467100.07 Д4	Арк.
						4
Зм.	Арк.	№ докум.	Підпис	Дата		

```

while len(data) < payload_size:
    received = connection.recv(4096)
    if received == b"":
        connection.close()
        self.__used_slots -= 1
        break_loop = True
        break
    data += received

if break_loop:
    break

packed_msg_size = data[:payload_size]
data = data[payload_size:]

msg_size = struct.unpack(">L", packed_msg_size)[0]

while len(data) < msg_size:
    data += connection.recv(4096)

frame_data = data[:msg_size]
data = data[msg_size:]

frame = pickle.loads(
    frame_data, fix_imports=True, encoding="bytes")
img1 = cv2.imdecode(frame, cv2.IMREAD_COLOR)
succ, img = cap.read()

markerSize = 6
totalMarkers = 250

```

					ІАЛЦ.467100.07 Д4	Арк.
						5
Зм.	Арк.	№ докум.	Підпис	Дата		

```

img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
key = getattr(aruco, 'DICT_6X6_250')
aruco_dict = aruco.Dictionary_get(key)
aruco_param = aruco.DetectorParameters_create()

res_boxes, res_ids, rej = aruco.detectMarkers(
    img_gray, aruco_dict, parameters=aruco_param)
my_polygon = [0, 0, 0, 0]
if res_boxes:
    polygon_ids = [[i[0][0], k[0]]
                    for i, k in zip(res_boxes, res_ids)]
    print(polygon_ids)
    if len(polygon_ids) == 4:
        for i in polygon_ids:
            if i[1] == 0:
                my_polygon[3] = i[0]
            elif i[1] == 1:
                my_polygon[2] = i[0]
            elif i[1] == 2:
                my_polygon[0] = i[0]
            elif i[1] == 3:
                my_polygon[1] = i[0]
        list1 = []
        list2 = []
        count = 0
        for i in my_polygon:
            if count == 3 and i[0] < my_polygon[1][0]:
                list1.insert(0, i)
            else:
                list1.append(i)

```

					ІАЛЦ.467100.07 Д4	Арк.
						6
Зм.	Арк.	№ докум.	Підпис	Дата		

```

list1.append(i)
if count == 3:
    if i[0] < my_polygon[1][0]:
        list2.insert(0, i)
        list2[2], list2[3] = list2[3], list2[2]
    else:
        list2.insert(2, i)
else:
    list2.append(i)

count += 1
h1, w1 = img1.shape[:2]

pts1 = np.float32([[0, 0], [0, h1], [w1, 0], [w1, h1]])
pts2 = np.float32(list2)

h, mask = cv2.findHomography(pts1, pts2,
cv2.RANSAC, 5.0)

height, width, channels = img.shape
im1Reg = cv2.warpPerspective(img1, h, (width,
height))

mask2 = np.zeros(img.shape, dtype=np.uint8)

roi_corners2 = np.int32(list1)

channel_count2 = img.shape[2]
ignore_mask_color2 = (255,) * channel_count2

```

					ІАЛЦ.467100.07 Д4	Арк.
						7
Зм.	Арк.	№ докум.	Підпис	Дата		

```

cv2.fillConvexPoly(mask2, roi_corners2,
ignore_mask_color2)

mask2 = cv2.bitwise_not(mask2)
masked_image2 = cv2.bitwise_and(img, mask2)
# Using Bitwise or to merge the two images
img = cv2.bitwise_or(im1Reg, masked_image2)
img = np.concatenate((img, img), axis=1)
cv2.imshow("test", img)
if cv2.waitKey(1) == ord(self.__quit_key):
    connection.close()
    self.__used_slots -= 1
    break

```

```

class REcievingClient:
    def __init__(self) -> None:
        pass

```

```

class StreamingClient:
    """
    Abstract class for the generic streaming client.

    Attributes
    -----

    Private:

    __host : str

```

					ІАЛЦ.467100.07 Д4	Арк.
						8
Зм.	Арк.	№ докум.	Підпис	Дата		

host address to connect to
 __port : int
 port to connect to
 __running : bool
 indicates if the client is already streaming or not
 __encoding_parameters : list
 a list of encoding parameters for OpenCV
 __client_socket : socket
 the main client socket

Methods

Private:

__client_streaming : main method for streaming the client data

Protected:

_configure : sets basic configurations (overridden by child classes)

_get_frame : returns the frame to be sent to the server (overridden by child classes)

_cleanup : cleans up all the resources and closes everything

Public:

start_stream : starts the client stream in a new thread

""""

					ІАЛЦ.467100.07 Д4	Арк.
						9
Зм.	Арк.	№ докум.	Підпис	Дата		

```
def __init__(self, host, port):
    """
    Creates a new instance of StreamingClient.

    Parameters
    -----

    host : str
        host address to connect to
    port : int
        port to connect to
    """
    self.__host = host
    self.__port = port
    self._configure()
    self.__running = False
    self.__client_socket = socket.socket(
        socket.AF_INET, socket.SOCK_STREAM)
```

```
def _configure(self):
    self.__encoding_parameters =
[int(cv2.IMWRITE_JPEG_QUALITY), 90]
```

```
def _get_frame(self):
    """
    Basic function for getting the next frame.
```

Returns

					ІАЛЦ.467100.07 Д4	Арк.
						10
Зм.	Арк.	№ докум.	Підпис	Дата		

frame : the next frame to be processed (default = None)

"""

return None

def _cleanup(self):

cv2.destroyAllWindows()

def __client_streaming(self):

self.__client_socket.connect((self.__host, self.__port))

while self.__running:

frame = self._get_frame()

result, frame = cv2.imencode(

'jpg', frame, self.__encoding_parameters)

data = pickle.dumps(frame, 0)

size = len(data)

try:

self.__client_socket.sendall(struct.pack('>L', size) + data)

except ConnectionResetError:

self.__running = False

except ConnectionAbortedError:

self.__running = False

except BrokenPipeError:

self.__running = False

self._cleanup()

def start_stream(self):

if self.__running:

					ІАЛЦ.467100.07 Д4	Арк.
						11
Зм.	Арк.	№ докум.	Підпис	Дата		

```

        print("Client is already streaming!")
    else:
        self.__running = True
        client_thread = threading.Thread(target=self.__client_streaming)
        client_thread.start()

```

```

def stop_stream(self):
    if self.__running:
        self.__running = False
    else:
        print("Client not streaming!")

```

```

class CameraClient(StreamingClient):

```

```

    """

```

```

    Class for the camera streaming client.

```

```

    Attributes

```

```

    -----

```

```

    Private:

```

```

        __host : str
            host address to connect to
        __port : int
            port to connect to
        __running : bool
            indicates if the client is already streaming or not
        __encoding_parameters : list
            a list of encoding parameters for OpenCV

```

					ІАЛЦ.467100.07 Д4	Арк.
						12
Зм.	Арк.	№ докум.	Підпис	Дата		

__client_socket : socket
the main client socket

__camera : VideoCapture
the camera object

__x_res : int
the x resolution

__y_res : int
the y resolution

Methods

Protected:

__configure : sets basic configurations

__get_frame : returns the camera frame to be sent to the server

__cleanup : cleans up all the resources and closes everything

Public:

start_stream : starts the camera stream in a new thread

"""

def __init__(self, host, port, x_res=1024, y_res=576):

"""

Creates a new instance of CameraClient.

Parameters

					ІАЛЦ.467100.07 Д4	Арк.
						13
Зм.	Арк.	№ докум.	Підпис	Дата		

```

host : str
    host address to connect to
port : int
    port to connect to
x_res : int
    the x resolution
y_res : int
    the y resolution
"""
self.__x_res = x_res
self.__y_res = y_res
self.__camera = cv2.VideoCapture(0)
super(CameraClient, self).__init__(host, port)

```

```

def _configure(self):
    """
    Sets the camera resolution and the encoding parameters.
    """
    self.__camera.set(3, self.__x_res)
    self.__camera.set(4, self.__y_res)
    super(CameraClient, self)._configure()

```

```

def _get_frame(self):
    """
    Gets the next camera frame.

    Returns
    -----

```

frame : the next camera frame to be processed

"""

ret, frame = self.__camera.read()

return frame

def _cleanup(self):

"""

Cleans up resources and closes everything.

"""

self.__camera.release()

cv2.destroyAllWindows()

class VideoClient(StreamingClient):

"""

Class for the video streaming client.

Attributes

Private:

__host : str

host address to connect to

__port : int

port to connect to

__running : bool

indicates if the client is already streaming or not

__encoding_parameters : list

a list of encoding parameters for OpenCV

					ІАЛЦ.467100.07 Д4	Арк.
						15
Зм.	Арк.	№ докум.	Підпис	Дата		

__client_socket : socket
the main client socket

__video : VideoCapture
the video object

__loop : bool
boolean that decides whether the video shall loop or not

Methods

Protected:

__configure : sets basic configurations

__get_frame : returns the video frame to be sent to the server

__cleanup : cleans up all the resources and closes everything

Public:

start_stream : starts the video stream in a new thread

"""

def __init__(self, host, port, video, loop=True):

"""

Creates a new instance of VideoClient.

Parameters

host : str

					ІАЛЦ.467100.07 Д4	Арк.
						16
Зм.	Арк.	№ докум.	Підпис	Дата		

```

        host address to connect to
    port : int
        port to connect to
    video : str
        path to the video
    loop : bool
        indicates whether the video shall loop or not
    """
    self.__video = cv2.VideoCapture(video)
    self.__loop = loop
    super(VideoClient, self).__init__(host, port)

def _configure(self):
    """
    Set video resolution and encoding parameters.
    """
    self.__video.set(3, 1024)
    self.__video.set(4, 576)
    super(VideoClient, self)._configure()

def _get_frame(self):
    """
    Gets the next video frame.

    Returns
    -----

    frame : the next video frame to be processed
    """
    ret, frame = self.__video.read()

```

					ІАЛЦ.467100.07 Д4	Арк.
						17
Зм.	Арк.	№ докум.	Підпис	Дата		

```
return frame
```

```
def _cleanup(self):
```

```
    """
```

```
    Cleans up resources and closes everything.
```

```
    """
```

```
    self.__video.release()
```

```
    cv2.destroyAllWindows()
```

```
class ScreenShareClient(StreamingClient):
```

```
    """
```

```
    Class for the screen share streaming client.
```

```
Attributes
```

```
-----
```

```
Private:
```

```
    __host : str
```

```
        host address to connect to
```

```
    __port : int
```

```
        port to connect to
```

```
    __running : bool
```

```
        inicates if the client is already streaming or not
```

```
    __encoding_parameters : list
```

```
        a list of encoding parameters for OpenCV
```

```
    __client_socket : socket
```

```
        the main client socket
```

```
    __x_res : int
```

					ІАЛІЦ.467100.07 Д4	Арк.
						18
Зм.	Арк.	№ докум.	Підпис	Дата		

the x resolution
__y_res : int
the y resolution

Methods

Protected:

_get_frame : returns the screenshot frame to be sent to the server

Public:

start_stream : starts the screen sharing stream in a new thread

"""

def __init__(self, host, port, x_res=1024, y_res=576):

"""

Creates a new instance of ScreenShareClient.

Parameters

host : str

host address to connect to

port : int

port to connect to

x_res : int

the x resolution

					ІАЛЦ.467100.07 Д4	Арк.
						19
Зм.	Арк.	№ докум.	Підпис	Дата		

```

        y_res : int
            the y resolution
        """
        self.__x_res = x_res
        self.__y_res = y_res
        super(ScreenShareClient, self).__init__(host, port)

    def _get_frame(self):

        screen = pyautogui.screenshot()
        frame = np.array(screen)
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        frame = cv2.resize(frame, (self.__x_res, self.__y_res),
                            interpolation=cv2.INTER_AREA)

        return frame

from vidstream import StreamingServer
import threading
import socket
print(socket.gethostname(socket.gethostname()))
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((socket.gethostname(), 1024))
s.send(bytes(socket.gethostname(socket.gethostname()), "utf-8"))
print('sended')

host = socket.gethostname(socket.gethostname())

reciever = StreamingServer(host, 8081)

```

					ІАЛЦ.467100.07 Д4	Арк.
						20
Зм.	Арк.	№ докум.	Підпис	Дата		

```
t = threading.Thread(target=reciever.start_server())
t.start()
```

```
while input("") != 'STOP':
    continue
reciever.stop_server()
```

```
from vidstream import ScreenShareClient
import threading
import socket
```

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((socket.gethostname(), 1024))
s.listen(5)
msg = ""
while True:
    clt, adr = s.accept()
    print("connection established")
    msg = clt.recv(1024)
    print(msg.decode("utf-8"))
    if clt:
        break
s.close()
```

```
sender = ScreenShareClient(msg, 8081)
```

```
t = threading.Thread(target=sender.start_stream())
t.start()
while input("") != 'STOP':
```

					ІАЛЦ.467100.07 Д4	Арк.
						21
Зм.	Арк.	№ докум.	Підпис	Дата		

continue

sender.stop_stream

					ІАЛЦ.467100.07 Д4	Арк.
						22
Зм.	Арк.	№ докум.	Підпис	Дата		

