ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ

метадані

Назва організації
Kharkiv National University of Radio Electronics
Заголовок
2025_М_ПІ_ІПЗм_23_1_Первєєва_Д_О_скорочений
Автор — Науковий керівник / Експерт
Первєєва Дар'я ОлександрівнаОлена Олійник
підрозділ
каф. ПІ

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

25	12816	102397
Довжина фрази для коефіцієнта подібності 2	Кількість слів	Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		2
Інтервали		0
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)		1

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	http://meda.nure.ua/wp-content/uploads/2021/11/zayava_zgoda-2021.docx	27 0.21 %
2	https://software.nure.ua/wp-content/uploads/2024/01/dyplom-mag-7.dotm.docx	23 0.18 %
3	https://software.nure.ua/wp-content/uploads/2024/01/mv_2023-end.pdf	17 0.13 %
4	https://software.nure.ua/wp-content/uploads/2024/01/dyplom-mag-7.dotm.docx	12 0.09 %
5	https://pubmed.ncbi.nlm.nih.gov/36583912/	10 0.08 %

6	http://meda.nure.ua/wp-content/uploads/2021/11/zayava_zgoda-2021.docx	7 0.05 %
7	123-М-2024-ПАВЛЮК ОВ [redacted] 11/25/2024 Ukrainian national aviation university (ФКНТ Кафедра інтелектуальних кібернетичних систем)	7 0.05 %
8	https://software.nure.ua/wp-content/uploads/2024/01/dyplom-mag-7.dotm.docx	7 0.05 %
9	2024_M_IMC_KP_ІПЗм-23-1_Філіпенко_А_В_скорочений [redacted] 6/4/2024 Kharkiv National University of Radio Electronics (Харківський національний університет радіоелектроніки)	6 0.05 %
10	123-М-2024-ПАВЛЮК ОВ [redacted] 11/25/2024 Ukrainian national aviation university (ФКНТ Кафедра інтелектуальних кібернетичних систем)	5 0.04 %

з бази даних RefBooks (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з домашньої бази даних (0.05 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	2024_M_IMC_KP_ІПЗм-23-1_Філіпенко_А_В_скорочений [redacted] 6/4/2024 Kharkiv National University of Radio Electronics (Харківський національний університет радіоелектроніки)	6 (1) 0.05 %

з програми обміну базами даних (0.09 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	123-М-2024-ПАВЛЮК ОВ [redacted] 11/25/2024 Ukrainian national aviation university (ФКНТ Кафедра інтелектуальних кібернетичних систем)	12 (2) 0.09 %

з Інтернету (0.80 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://software.nure.ua/wp-content/uploads/2024/01/dyplom-mag-7.dotm.docx	42 (3) 0.33 %
2	**http://meda.nure.ua/wp-content/uploads/2021/11/zayava_zgoda-2021.docx**	**34 (2) 0.27 %**
3	https://software.nure.ua/wp-content/uploads/2024/01/my_2023-end.pdf	17 (1) 0.13 %
4	https://pubmed.ncbi.nlm.nih.gov/36583912/	10 (1) 0.08 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)

Завідувачу кафедри ПІ
проф. Кирилу СМЕЛЯКОВУ

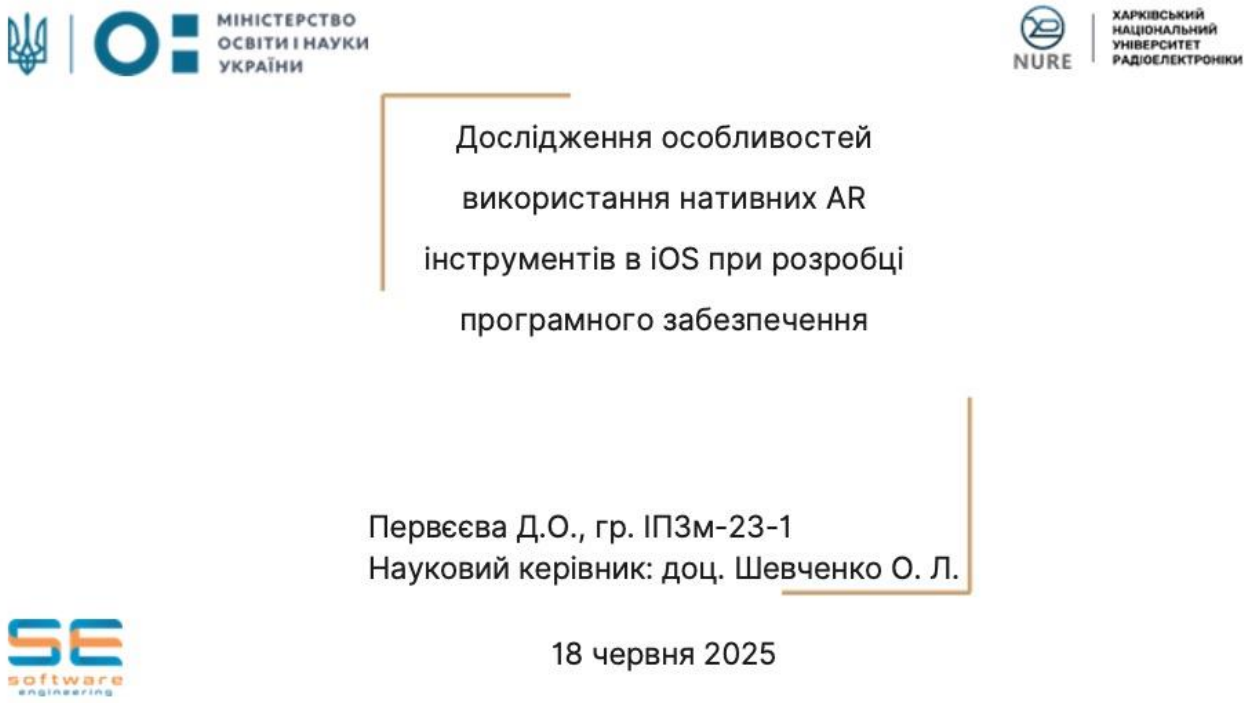
ЗАЯВА **щодо самостійності виконання кваліфікаційної роботи та можливості її публікації (та/або публікації анотації кваліфікаційної роботи) в електронному архіві відкритого доступу ElAr KhNURE Я,** Первєєва Дар'я Олександрівна, **здобувач вищої освіти на другому (магістерському) рівні** вищої освіти академічної групи ІПЗм-23-1 кафедра

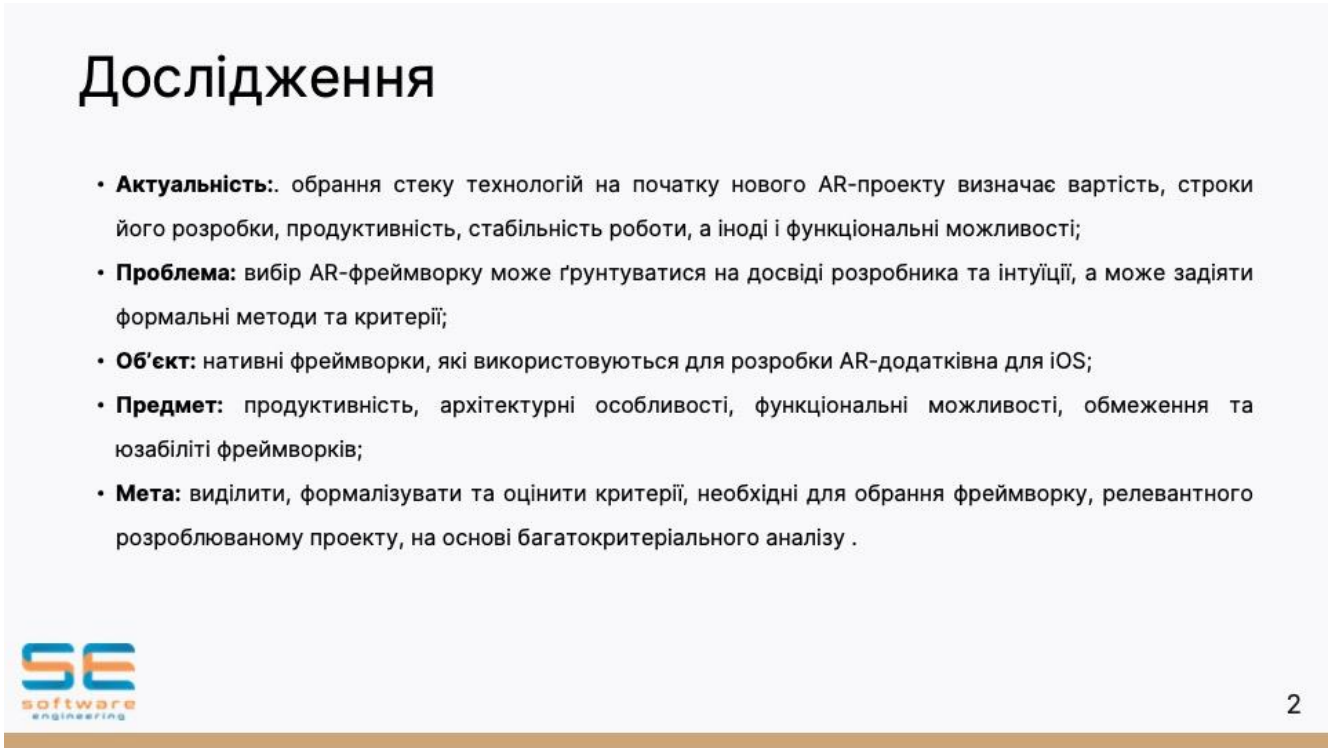
програмної інженерії, **заявляю: моя кваліфікаційна робота на тему «Дослідження** особливостей використання нативних AR інструментів в iOS при розробці програмного забезпечення», **що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в** репозиторії "ElArKhNURE". Погоджуюся з авторським договором, відповідно до Положення про репозиторій ХНУРЕ **"ElArKhNURE". Всі запозичення з друкованих та електронних джерел мають відповідні посилання. Я** ознайомлена з вимогами академічної доброчесності, згідно з якими **виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.**

Дата 07.06.2025 Підпис

ДОДАТОК Б

Презентаційний матеріал до кваліфікаційної роботи





Огляд літератури

- **Переваги фреймворків:**
  - ARKit – стабільний трекінг і точне позиціонування.
  - SceneKit – розширена кастомізація, зручна робота з 3D-графікою.
  - RealityKit – простота використання, сучасні рендеринг-технології, хороша оптимізація під мобільні пристрої.
- **Обмеження:**
  - SceneKit – менш ефективний у складних AR-сценах, немає підтримки сучасних шейдерів.
  - RealityKit – обмежена підтримка кастомних ефектів і шейдерів.
  - ARKit – вимагає зовнішнього фреймворку для рендерингу.
- **Прогалини в дослідженнях:**
  - Недостатньо експериментальних даних щодо кількісної продуктивності.
  - Відсутні комплексні порівняння SceneKit і RealityKit у реальних AR-сценаріях.
  - Не досліджено ресурсне навантаження (CPU, GPU) при різних типах сцен.

Постановка задачі

- **Проаналізувати** сучасні підходи до використання фреймворків ARKit, SceneKit і RealityKit у розробціAR-додатків на iOS та досвід їх застосування;
- **Визначити** набір якісних і кількісних критеріїв, за якими можна здійснювати об'єктивне порівняння фреймворків, розробити шкали оцінювання та оцінити кожен з фреймворків за визначеними якісними критеріями;
- **Розробити** власну програму з інтеграцією SceneKit і RealityKit для проведення експериментів у типових сценаріях AR-додатків, що відображають реальні навантаження;
- **Реалізувати** тестові сцени з різним рівнем складності: статичні, анімовані, з фізичними ефектами та взаємодією в режимі реального часу;
- **Провести** експериментальне дослідження з вимірюванням продуктивності і стабільності візуалізації при зміні параметрів сцени та оцінити кожен з фреймворків за обраними критеріями;
- **Сформувати** рекомендації та запропонувати підхід до (на основі багатокритеріального аналізу) вибору фреймворку залежно від технічних потреб проєкту .

Опис інструментів

ARKit

- Виявлення площин, трекінг руху, підтримка LiDAR
- ARAnchors, ARWorldMap, інтеграція з сенсорами
- Не має власного рушія для рендерингу сцен

SceneKit

- Рендеринг 3D-графіки, анімацій та фізики
- Працює з AR лише у зв'язці з ARKit
- Гнучке налаштування, але потребує більше ресурсів

RealityKit

- AR-фреймворк нового покоління, для створення продуктивного фотореалістичного контенту
- Вбудована фізика, просторовий звук, підтримка LiDAR
- Простий API

Об'єкт порівняння

- SceneKit та RealityKit: рендеринг сцени у зв'язці з ARKit
- ARKit як основа для обох рішень (трекінг, якорі, навігація)

5

Ключові критерії оцінювання фреймворків

Оцінювання продуктивності фреймворків здійснюється за такими критеріями:

- FPS (Frames Per Second): оцінка плавності анімацій і стабільності сцени;
- Завантаженість CPU: обробка логіки, анімацій, фізики та сенсорних даних;
- Використання GPU: рендеринг сцени, освітлення, текстури, відбиття;
- Стабільність при навантаженні: перевірка продуктивності при великій кількості об'єктів.

6

Додаткові критерії оцінювання фреймворків

Підтримка форматів 3D-моделей: GLTF, OBJ, FBX, USDZ, USD

Простота використання: документація, API, інтеграції, легкість побудови сцени у коді.

Обмеження: сумісність, масштабованість.

Анімація та фотореалізм: якість сцени, природність взаємодії, анімація, фотореалізм.

Фізичні симуляції: правдоподібна поведінка об'єктів.

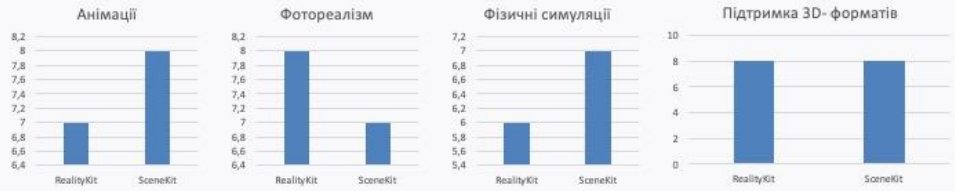
Налаштування та кастомізація: гнучке налаштування матеріалів, світла, шейдерів.

Ступінь інтеграції з AR операціями: зручність використання в AR-проектах.

Формалізація критеріїв оцінювання

Введено шкалу від 0 до 10 для якісних критеріїв:

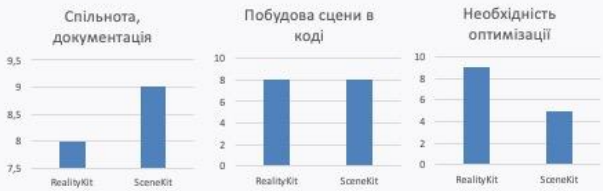
Візуальні можливості:



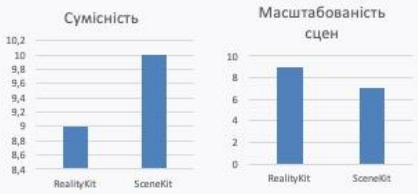
Налаштування та кастомізація:

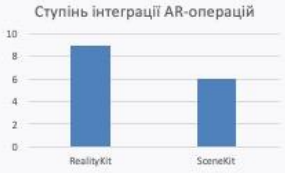
Формалізація критеріїв оцінювання

Простота використання:



Обмеження:





Бали виставлені з урахуванням експертних оцінок 5 експертів. Узгодження проводилося методом Delphi.

9

Проведення експерименту

- **Прототип середовища:**
  - AR-застосунок, реалізований на Swift з використанням SceneKit та RealityKit.
  - Експериментальне середовище запущене на iPhone 15 з iOS 18, 8 ГБ ОЗУ.
- **Сценарії тестування:**
  - Додавання до сцени 100 віртуальних об’єктів із різними розмірами й кольорами.
  - Запуск анімації обертання для кожного з об’єктів .
  - Поступове видалення об’єктів для оцінки стабільності.
  - Повторення кожного тесту кілька разів для перевірки надійності результатів.
- **Параметри оцінки:**
  - FPS (частота кадрів) – у реальному часі.
  - Завантаження CPU та GPU під час анімацій і взаємодії.
  - Аналіз реакції системи на зміну кількості об’єктів.

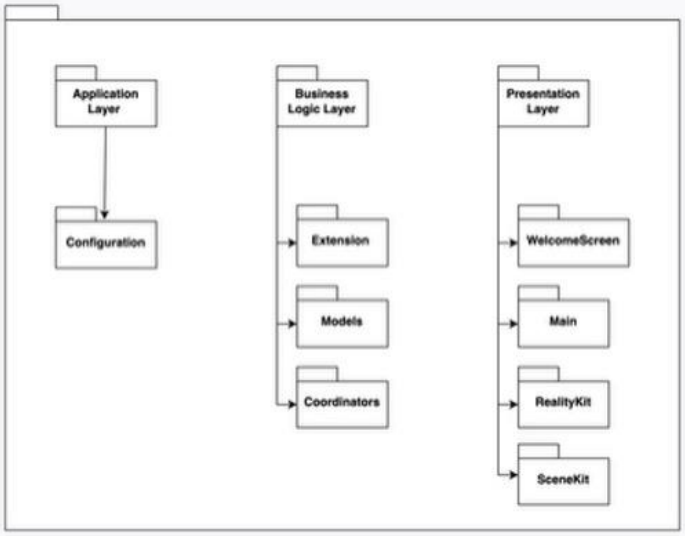
10

Опис програмного забезпечення

- **Процес розробки**
  - Створення тестового AR-застосунку на Swift з двома реалізаціями сцени: SceneKit та RealityKit;
  - Налаштування сцен з 3D-об'єктами, анімаціями та фізикою;
  - Розробка логіки для запуску, відображення та видалення об'єктів;
  - Підготовка механізмів фіксації FPS, CPU та GPU-метрик;
  - Проведення повторюваних експериментів із фіксацією даних;

- **Вибрані мови програмування та фреймворки**
  - Swift для реалізації;
  - ARKit, SceneKit для візуалізації сцени;
  - RealityKit для альтернативної реалізації;
  - Xcode Instruments для збору параметрів навантаження та продуктивності.

Діаграма пакетів



Показники продуктивності

Показник	Діапазон	Опис
Середнє значення FPS	0-60	Визначає плавність роботи додатка
Епізодичні просідання FPS	0-60	Падіння FPS, що негативно впливає на користувацький досвід
Початкове просідання FPS	0-60	Мінімальне значення FPS під піковим навантаженням
Максимальна частота CPU, ГГц	1-3.2	Наскільки сильно процесор «розганяється» під навантаженням; побічно вказує на рівень використання CPU
Максимальне завантаження GPU, %	0-100	Відсоток використання GPU під максимальним навантаженням

Візуальні можливості фреймворків

Показник	SceneKit	RealityKit	Опис
Формати 3D-файлів (GLTF, OBJ, FBX, USDZ, USD)	8	8	
Анімації	8	7	SceneKit не підтримує анімацію в реальному часі, RealityKit – інверсну кінематику та стани анімацій
Фотореалізм	7	8	SceneKit потребує ручного налаштування, RealityKit має автоматичне освітлення
Фізичні симуляції	7	6	SceneKit має базову фізику, RealityKit – прості тілесні взаємодії
Середнє значення	7.5	7.25	

Налаштування та кастомізація

Показник	SceneKit	RealityKit	Опис
Оптимізація продуктивності	8	5	SceneKit підтримує LOD та профілювання, RealityKit – ECS підхід
Кастомізація	9	5	SceneKit підтримує шейдери та PBR, RealityKit – базове налаштування
Середнє значення	8.5	5	

15

Обмеження фреймворків

Показник	SceneKit	RealityKit	Опис
Сумісність	10	9	SceneKit – для всіх версій iOS/macOS/visionOS; RealityKit – з iOS 13+ / macOS 10.15+
Масштабованість складних сцен	7	9	SceneKit потребує оптимізації; RealityKit – ефективний завдяки Metal
Середнє значення	8.5	9	

16

Простота використання

Показник	SceneKit	RealityKit	Опис
Спільнота, документація	9	8	Обидва фреймворки мають гарну документацію
Структура та побудова сцени	8	8	В обох фреймворках сцена створювалась вручну через код
Необхідність оптимізації	5	9	SceneKit потребує ручного налаштування, RealityKit – автоматична оптимізація
Середнє значення	7.5	8.3	

Підсумкові результати експерименту

Показник	SceneKit	RealityKit
Середнє значення FPS	58	54
Епізодичні просідання FPS	4	9
Початкове просідання FPS	55	9
Максимальна частота CPU, ГГц	1.68	2.13
Максимальне завантаження GPU, %	18	53

Аналіз результатів

FPS:

SceneKit підтримує стабільно високий FPS, з короткими просіданнями під час додавання об'єктів.

RealityKit має сильні падіння в момент ініціалізації сцени, але згодом стабілізується.

Завантаження ресурсів:

SceneKit демонструє низьке завантаження GPU і помірне використання CPU.

RealityKit створює високе навантаження як на CPU, так і на GPU.

Висновок:

SceneKit — стабільніший при роботі на обмежених ресурсах.

RealityKit — краща графіка і фізика, але з вищим навантаженням.

Нормалізовані оцінки:

Показник	SceneKit	RealityKit
Середнє значення FPS	9.7	9
Епізодичні просідання FPS	0.67	1.5
Початкове просідання FPS	9.17	1.5
Максимальна частота CPU, ГГц	3.09	5.14
Максимальне завантаження GPU, %	1.8	5.3

Значення були нормалізовані до діапазону [0-10] за формулою

$$x_{norm} = \frac{x_i - x_{min}}{x_{max} - x_{min}} * 10$$

Зведена порівняльна оцінка фреймворків

Критерій	SceneKit	RealityKit
Продуктивність	4.89	4.49
Графічні можливості	7.5	7.25
Інтеграція з AR	6	9
Налаштування та кастомізація	8.5	5
Простота використання	7.5	8.3
Обмеження	8.5	9

Багатокритеріальний аналіз: Застосунок 1 - каталог меблів

Сценарій:

AR-додаток-каталог меблевої продукції з можливістю сканування розміру вільного простору в кімнаті та розміщення меблів з каталогу, створюється невеликою стартап-командою.

Вимоги:

фотореалізм, якісне відстеження поверхонь, взаємодія з LiDAR для точного (до см.) сканування розмірів простору, висока продуктивність та оптимізація, робота з жестами, простота розробки.

21

Порівняння SceneKit + ARKit та RealityKit для каталогу меблів

Ключові вагові коефіцієнти:

Інтеграція з AR (0,25), Продуктивність (0,25), Простота використання (0,2)

Результати:

SceneKit + ARKit – **6.67**

RealityKit – **7.2**

Висновок:

RealityKit є кращим вибором завдяки вбудованому фотореалізму, якості вимірювання простору, продуктивності та простоті розробки

Критерій	Вага	SceneKit + ARKit		RealityKit	
		c_i	$c_i * w_i$	c_i	$c_i * w_i$
Продуктивність	0,25	4.89	1.22	4.49	1.12
Графічні можливості	0,1	7.5	0.75	7.25	0.725
Інтеграція з AR	0,25	6	1.5	9	2.25
Налаштування та кастомізація	0,1	8.5	0.85	5	0.5
Простота використання	0,2	7.5	1.5	8.2	1.7
Обмеження	0,1	8.5	0.85	9	0.9
Загальна оцінка			6.67		7.2

22

Багатокритеріальний аналіз: Застосунок 2 - фізична лабораторія

Сценарій:

навчальна фізична лабораторія, яка моделює віртуальні механічні установки, щоб досліджувати та експериментувати з точними фізичними взаємодіями, регулюючи фізичні характеристики компонентів, розробляється досвідченою у сфері програмування 3D-графіки командою .

Вимоги:

точність та реалістичність фізичних симуляцій, можливість кастомізації фізичних властивостей компонентів, детальна не обов'язково візуально реалістична візуалізація механічних компонентів .

23

Порівняння SceneKit + ARKit та RealityKit для технічного застосунку

Ключові вагові коефіцієнти:

Налаштування та кастомізація (0,3), Графічні можливості (0,2), Інтеграція з AR (0,15)

Результати:

SceneKit + ARKit – **6.88**

RealityKit – **6.53**

Висновок:

SceneKit виявився оптимальним, бо задача потребує кастомізації фізичних взаємодій.

Критерій	Вага	SceneKit + ARKit		RealityKit	
		c_i	$c_i * w_i$	c_i	$c_i * w_i$
Продуктивність	0,25	4.89	1.22	4.49	1.12
Графічні можливості	0,2	7.5	1.5	7.25	2.18
Інтеграція з AR	0,15	6	0.9	9	1.35
Налаштування та кастомізація	0,3	8.5	1.7	5	1
Простота використання	0,05	6.7	0.38	8.5	0.43
Обмеження	0,05	8.5	0.43	9	0.45
Загальна оцінка			6.88		6.53

24

Рекомендації та перспективи

Рекомендації за результатами:

- RealityKit рекомендовано використовувати для проєктів, у яких пріоритетом є точність AR-взаємодії, висока якість і реалістичність графіки, а також швидкість і простота розробки;
- SceneKit краще підходить для задач із високими вимогами до гнучкості кастомізації графіки та фізичних взаємодій.

Особливості MCDA-аналізу:

У випадку наявності must have властивостей у альтернатив, між якими необхідно зробити вибір, такі властивості необхідно перевіряти до початку проведення MCDA.

Публікація результатів

Результати роботи були прийняті до участі у міжнародній науковій конференції – XVIII Міжнародна науково-практична конференція магістрантів та аспірантів «Теоретичні та практичні дослідження молодих вчених», Харків, НТУ ХПІ, 19-22 листопада 2024 р

УДК 004.4'2

ПОРІВНЯННЯ ХАРАКТЕРИСТИК ІНСТРУМЕНТІВ ДЛЯ РОБОТИ З ДОПОВНЕНОЮ РЕАЛЬНІСТЮ НА IOS

Д.О. Перестюк[1], М.С. Широкопетлюк[2]

[1] магістрант кафедри інженерії програмного забезпечення, ХНУРЕ, Харків, Україна
[2] старший викладач кафедри програмної інженерії, ХНУРЕ, Харків, Україна

Опанування AR стає критично навичкою для інженерів, адже інтерактивний контент стає нормою. Розробники стикаються з вибором серед численних фреймворків для AR, кожен із яких пропонує унікальні можливості. Вибір оптимального інструменту залежить від таких ключових критеріїв, як продуктивність, плавність рендерингу та ефективне використання ресурсів, зокрема CPU і GPU.

Мета цього дослідження - аналіз та порівняння інструментів ARKit, SceneKit і RealityKit з точки зору продуктивності рендерингу сцен, оптимізації ресурсів та вибору найбільш підходящого інструменту для різних типів проектів. Дослідження дозволить продукти, які фреймворки забезпечують кращу стабільність і плавність рендерингу при використанні складних 3D-об'єктів та анімацій, а також їх вплив на використання CPU та GPU мобільного пристрою.

Для досягнення цієї мети було поставлено такі завдання:

- Провести аналіз відомих інструментів для роботи з доповненою реальністю на iOS: ARKit, SceneKit і RealityKit;
- Визначити критерії порівняння фреймворків та створити сценарій тестування;
- Написати додаток з реалізацією функціоналу на основі сценарію;
- Провести експерименти для збору даних, а саме FSP, CPU, GPU під час виконання типових операцій з 3D-об'єктами в кожному з фреймворків;
- Надати рекомендації щодо використання того чи іншого інструменту.

Предметом дослідження є інструменти для розробки AR-додатків на iOS: ARKit, SceneKit і RealityKit. Об'єктом дослідження виступає продуктивність зазначених бібліотек і їх здатність ефективно використовувати ресурси мобільних пристроїв, такі як CPU і GPU. Особливу увагу буде приділено стабільності рендерингу, рівню використання ресурсів під час одночасного виконання анімацій та взаємодії з 3D-об'єктами.

ARKit - фреймворк від Apple для створення доповненої реальності (AR) на iOS, що забезпечує виявлення поверхонь, визначення позиції пристрою та глибини для інтеграції віртуальних об'єктів у реальний простір.

SceneKit - гнучкий інструмент для рендерингу 3D-графіки, що підтримує створення складних тривимірних сцен, включаючи матеріали, світло, камери, анімації та фізичні симуляції. SceneKit не є основним для AR, але використовується для візуалізації 3D-контенту разом із ARKit.

RealityKit - новітній фреймворк для AR від Apple, випущений у 2019 році, пропонує реалістичний рендеринг і фізичну взаємодію об'єктів із навколишнім середовищем. Підтримка LiDAR дозволяє RealityKit створювати точні сцени для сучасних пристроїв.

Розробка сценаріїв для тестування продуктивності AR-додатка на iOS є важливим етапом, що дозволяє оцінити ефективність використання ресурсів та стабільність додатка під навантаженням. Методи є моделювання типових умов використання додатка для виявлення "вузьких місць" у продуктивності, які можуть впливати на

193

Публікація результатів

Результати роботи були апробовані на міжнародній конференції – «16th International Scientific and Practical Conference «Environment. Technology. Resources» hosted by RTU Rezekne Academy, Rezekne, Latvia, June 19-20, 2025»

A Comparative Analysis of Native iOS Frameworks for Developing Graphics Scenes in Augmented Reality Applications

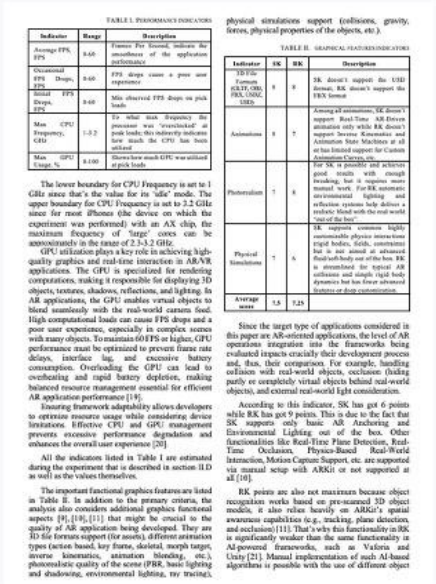


TABLE I. PERFORMANCE INDICATORS

Indicator	Range	Description
Average FPS, FPS	0-60	Frames Per Second, indicate the smoothness of the application performance
Occasional FPS Drops, FPS	0-60	FPS drops cause a poor user experience
Initial FPS Drops, FPS	0-60	Min observed FPS drops on pick loads
Max CPU Frequency, GHz	1-3.2	To what max frequency the processor was "overclocked" at peak loads; this indirectly indicates how much the CPU has been utilized
Max GPU Usage, %	0-100	Shows how much GPU was utilized at pick loads

The lower boundary for CPU Frequency is set to 1 GHz since that's the value for its 'idle' mode. The upper boundary for CPU Frequency is set to 3.2 GHz since for most iPhones (the device on which the experiment was performed) with an AX chip, the maximum frequency of 'large' cores can be approximately in the range of 2.3-3.2 GHz.

GPU utilization plays a key role in achieving high-quality graphics and real-time interaction in AR/VR applications. The GPU is specialized for rendering computations, making it responsible for displaying 3D objects, textures, shadows, reflections, and lighting. In AR applications, the GPU enables virtual objects to blend seamlessly with the real-world camera feed. High computational loads can cause FPS drops and a poor user experience, especially in complex scenes with many objects. To maintain 60 FPS or higher, GPU performance must be optimized to prevent frame rate delays, interface lag, and excessive battery consumption. Overloading the GPU can lead to overheating and rapid battery depletion, making balanced resource management essential for efficient AR application performance [19].

Ensuring framework adaptability allows developers to optimize resource usage while considering device limitations. Effective CPU and GPU management prevents excessive performance degradation and enhances the overall user experience [20].

All the indicators listed in Table I are estimated during the experiment that is described in section II.D as well as the values themselves.

The important functional graphics features are listed in Table II. In addition to the primary criteria, the analysis also considers additional graphics functional aspects [9], [10], [11] that might be crucial to the quality of AR application being developed. They are 3D file formats support (for assets), different animation types (action based, key frame, skeletal, morph target, inverse kinematics, animation blending, etc.), photorealistic quality of the scene (PBR, basic lighting and shadowing, environmental lighting, ray tracing), physical simulations support (collisions, gravity, forces, physical properties of the objects, etc.).

TABLE II. GRAPHICAL FEATURES INDICATORS

Indicator	SK	RK	Description
3D File Formats (GLTF, OBJ, FBX, USDZ, USD)	8	8	SK doesn't support the USD format, RK doesn't support the FBX format
Animations	8	7	Among all animations, SK doesn't support Real-Time AR-Driven animation only while RK doesn't support Inverse Kinematics and Animation State Machines at all or has limited support for Custom Animation Curves, etc.
Photorealism	7	8	For SK is possible and achieves good results with enough tweaking, but it requires more manual work. For RK, automatic environmental lighting and reflection systems help deliver a realistic blend with the real-world "out of the box".
Physical Simulations	7	6	SK supports common highly customizable physics interactions (rigid bodies, fields, constraints) but is not aimed at advanced fluid/soft-body out of the box. RK is streamlined for typical AR collisions and simple rigid body dynamics but has fewer advanced features or deep customization.
Average score	5.5	7.25	

Since the target type of applications considered in this paper are AR-oriented applications, the level of AR operations integration into the frameworks being evaluated impacts crucially their development process and, thus, their comparison. For example, handling collision with real-world objects, occlusion (hiding partly or completely virtual objects behind real-world objects), and external real-world light consideration.

According to this indicator, SK has got 6 points while RK has got 9 points. This is due to the fact that SK supports only basic AR Anchoring and Environmental Lighting out of the box. Other functionalities like Real-Time Plane Detection, Real-Time Occlusion, Physics-Based Real-World Interaction, Motion Capture Support, etc. are supported via manual setup with ARKit or not supported at all [10].

RK points are also not maximum because object recognition works based on pre-scanned 3D object models, it also relies heavily on ARKit's spatial awareness capabilities (e.g., tracking, plane detection, and occlusion) [11]. That's why this functionality in RK is significantly weaker than the same functionality in AI-powered frameworks, such as Vuforia and Unity [21]. Manual implementation of such AI-based algorithms is possible with the use of different object

software engineering

Висновки

- Проведене комплексне дослідження фреймворків для розробки додатків доповненої реальності на платформі iOS;
- Виділено, формалізовано та оцінено критерії, достатні для обрання фреймворку, релевантного розроблюваному проекту;
- Для оцінки продуктивності порівнюваних фреймворків розроблене програмне забезпечення, яке дозволяє оцінити роботу фреймворків у різних режимах навантаження;
- Запропоновано підхід з використанням багатокритеріального аналізу для обрання фреймворку, релевантного до розроблюваного проєкту;
- Надано загальні рекомендації щодо використання обох фреймворків та особливостей проведення багатокритеріального аналізу
- У майбутньому перспективним напрямом досліджень є інтеграція AI-засобів у роботу фреймворків, зокрема детальне просторове розуміння, семантична сегментація сцени та розпізнавання об'єктів.

software engineering

Дякую за увагу!

ДОДАТОК В

Матеріали XVIII Міжнародної науково-практичної конференції магістрантів та аспірантів «Теоретичні та практичні дослідження молодих вчених», Харків, НТУ ХПІ, 19-22 листопада 2024 р


МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
„ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"

TALLINN UNIVERSITY OF TECHNOLOGY, ESTONIA

RIGA TECHNICAL UNIVERSITY, LATVIA

XVIII МІЖНАРОДНА
НАУКОВО-ПРАКТИЧНА КОНФЕРЕНЦІЯ
МАГІСТРАНТІВ ТА АСПІРАНТІВ
(19–22 листопада 2024 року)

Матеріали конференції

Харків 2024

Рисунок А.1 – Титульна сторінка збірника конференції (рисунок виконаний самостійно)

УДК 002

М43

Голова конференції – ректор НТУ «ХПІ» Є.І. Сокол.

Співголови конференції: Д. Вінніков (Естонія), І. Галкін (Латвія).

Члени програмного комітету: А.П. Марченко, Р.В. Кривобок, Д.О. Данильченко

Члени організаційного комітету: Р.П. Мигущенко, К.О. Мінакова, М.Д. Годлевський, В.В. Єпіфанов, Ю.І. Зайцев, А.В. Кіпенський, Н.С. Краснокутська, Д.А. Горовий, О.О. Ларін, І.М. Рищенко, Р.С. Томашевський, Г.С. Хрипунов.

Секретаріат конференції: О.С. Махонина, М.М. Козуля

М43 **XVIII Міжнародна** науково-практична конференція магістрантів та аспірантів «Теоретичні та практичні дослідження молодих вчених» (19–22 листопада 2024 року): матеріали конференції / за ред. проф. Є.І. Сокола. – Харків : НТУ «ХПІ», 2024. – 840

УДК 002

ISBN 978-617-05-0514-9

Рисунок А.2 – Друга сторінка збірника конференції (рисунок виконаний самостійно)

УДК 004.4`2

ПОРІВНЯННЯ ХАРАКТЕРИСТИК ІНСТРУМЕНТІВ ДЛЯ РОБОТИ З ДОПОВНЕНОЮ РЕАЛЬНІСТЮ НА IOS

***Д.О. Первєєва*[1]**, *М.С. Широкопетлєва*[2]

[1] *магістрант кафедри Інженерії програмного забезпечення, ХНУРЕ, Харків, Україна*
[2] *старший викладач кафедри програмної інженерії, ХНУРЕ, Харків, Україна*
daria.koshkina@nure.ua

Опанування AR стає критичною навичкою для інженерів, адже інтерактивний контент стає нормою. Розробники стикаються з вибором серед численних фреймворків для AR, кожен із яких пропонує унікальні можливості. Вибір оптимального інструменту залежить від таких ключових критеріїв, як продуктивність, плавність рендерингу та ефективне використання ресурсів, зокрема CPU і GPU.

Мета цього дослідження – аналіз та порівняння інструментів ARKit, SceneKit і RealityKit з точки зору продуктивності рендерингу сцен, оптимізації ресурсів та вибору найбільш підходящого інструменту для різних типів проектів. Дослідження дозволить зрозуміти, які фреймворки забезпечують кращу стабільність і плавність рендерингу при використанні складних 3D-об'єктів та анімацій, а також їх вплив на використання CPU та GPU мобільного пристрою.

Для досягнення цієї мети було поставлено такі завдання:

- Провести аналіз відомих інструментів для роботи з доповненою реальністю на iOS: ARKit, SceneKit і RealityKit;
- Визначити критерії порівняння фреймворків та створити сценарій тестування;
- Написати додаток з реалізацією функціоналу на основі сценарію;
- Провести експерименти для збору даних, а саме FSP, CPU, GPU під час виконання типових операцій з 3D-об'єктами в кожному з фреймворків;
- Надати рекомендації щодо використання того чи іншого інструменту.

Предметом дослідження є інструменти для розробки AR-додатків на iOS: ARKit, SceneKit і RealityKit. Об'єктом дослідження виступає продуктивність зазначених бібліотек і їх здатність ефективно використовувати ресурси мобільних пристроїв, такі як CPU і GPU. Особливу увагу буде приділено стабільності рендерингу, рівню використання ресурсів під час одночасного виконання анімацій та взаємодії з 3D-об'єктами.

ARKit – фреймворк від Apple для створення доповненої реальності (AR) на iOS, що забезпечує виявлення поверхонь, визначення позиції пристрою та глибини для інтеграції віртуальних об'єктів у реальний простір.

SceneKit – гнучкий інструмент для рендерингу 3D-графіки, що підтримує створення складних тривимірних сцен, включаючи матеріали, світло, камери, анімації та фізичні симуляції. SceneKit не є основним для AR, але використовується для візуалізації 3D-контенту разом із ARKit.

RealityKit – новітній фреймворк для AR від Apple, випущений у 2019 році, пропонує реалістичний рендеринг і фізично коректну взаємодію об'єктів із навколишнім середовищем. Підтримка LiDAR дозволяє RealityKit створювати точні сцени для сучасних пристроїв.

Розробка сценаріїв для тестування продуктивності AR-додатка на iOS є важливим етапом, що дозволяє оцінити ефективність використання ресурсів та стабільність додатка під навантаженням. Метою є моделювання типових умов використання додатка для виявлення "вузьких місць" у продуктивності, які можуть вплинути на

Рисунок А.3 – Перша сторінка тез конференції (рисунок виконаний самостійно)

користувацький досвід. Важливо врахувати не лише стандартні операції, а й специфіку фреймворків ARKit, SceneKit і RealityKit.

Дослідження продуктивності ARKit, SceneKit і RealityKit проводитеметься через спостереження за виконанням операцій з 3D-об'єктами у сцені, вимірюючи FPS, CPU та GPU. Основні критерії включають FPS (для плавного користувацького досвіду) [1], CPU (обробка графіки, анімацій, фізики об'єктів) [2] і GPU (ключовий для реалістичного рендерингу) [3]. Для тестування буде використовуватись Xcode Instruments – потужний інструмент для діагностики продуктивності, що дозволяє вимірювати FPS, CPU і GPU навантаження в реальному часі. Специфічні інструменти включатимуть Core Animation FPS (моніторинг стабільності FPS), Time Profiler (відстеження навантаження на CPU) та GPU Profiler (оцінка ефективності використання GPU, особливо для порівняння 3D-графіки SceneKit і фотореалізму RealityKit).

Сценарій тестування включає додавання 100 кубиків різного розміру та кольору для перевірки здатності системи рендерити багатошарову сцену. Наступний етап – одночасна анімація обертання об'єктів для перевірки обробки руху та стабільності FPS під навантаженням. Останній крок – поетапне видалення об'єктів для оцінки вивільнення ресурсів і відновлення продуктивності.

Цей сценарій забезпечує чіткі дані для порівняння продуктивності, стабільності та ефективності фреймворків, моделюючи реальні умови використання в додатках. Для цього створено додаток ARPerfTester, який реалізує тестовий сценарій для вимірювання продуктивності рендерингу, використання CPU та GPU, а також загальної ефективності фреймворків. Дослідження продуктивності ARKit, SceneKit і RealityKit надає розробникам рекомендації щодо вибору оптимального інструменту для AR-додатків на iOS. ARKit забезпечує стабільну частоту кадрів і ефективне використання ресурсів, що робить його ідеальним для мобільних ігор, онлайн-шопінгу та освітніх додатків. SceneKit підходить для рендерингу 3D-графіки та анімацій, особливо для 3D-ігор, де важлива візуальна складова без складної фізики чи інтеграції з реальним світом. RealityKit із фотореалістичною візуалізацією та фізичними взаємодіями потребує більше ресурсів, але найкраще підходить для програм, де важлива висока якість графіки, як у віртуальних шоурумах і програмах для туризму.

Аналіз цих фреймворків підкреслює важливість ефективного управління ресурсами GPU та CPU для продуктивності AR-додатків. Оптимізація дозволяє адаптувати додатки під специфіку пристрою, забезпечуючи плавний AR-досвід. Дослідження показало різні підходи до використання ресурсів: ARKit підтримує стабільний FPS та ефективність, оптимальний для інтеграції з реальним світом; SceneKit з низьким споживанням CPU підходить для 3D-графіки; RealityKit з фотореалістичною візуалізацією потребує значних ресурсів і часто має коливання FPS, що вимагає оптимізації. Подальші дослідження варто зосередити на покращенні рендерингу в RealityKit, комбінуванні ARKit і SceneKit та використанні LiDAR для підвищення точності інтеграції об'єктів у реальному середовищі.

Список літератури:

1. Boutsi, A.-M., Ioannidis, Ch., Verykokou, S. Multi-Resolution 3D Rendering for High-Performance Web AR / A.-M. Boutsi, Ch. Ioannidis, S. Verykokou // Sensors. – 2 023. – Vol. 23. – No. 15. – P. 6885. – DOI: https://doi.org/10.3390/s23156885

2. Hort, M., Kechagia, M., Sarro, F., Harman, M. A Survey of Performance Optimization for Mobile Applications / M. Hort, M. Kechagia, F. Sarro, M. Harman // IEEE Transactions on Software Engineering. – 2022. – Vol. 48. – No. 8. – P. 2879–2904. – DOI: 10.1109/TSE.2021.3071193.

3. Nusrat, F., Hassan, F., Zhong, H., Wang, X. How Developers Optimize Virtual Reality Applications: A Study of Optimization Commits in Open Source Unity Projects / F. Nusrat, F. Hassan, H. Zhong, X. Wang // IEEE/ACM 43rd International Conference on Software Engineering (ICSE). – Madrid, ES, 2021. – P. 473–485. – DOI: 10.1109/ICSE43902.2021.00052.

194

Рисунок А.4 – Друга сторінка тез конференції (рисунок виконаний самостійно)

ДОДАТОК Г

Матеріали конференції «16th International Scientific and Practical Conference «Environment. Technology. Resources» hosted by RTU Rezekne Academy, Rezekne, Latvia, June 19-20, 2025»

Environment. Technology. Resources. Rezekne, Latvia
Proceedings of the 16th International Scientific and Practical Conference. Volume X, XX-XX

A Comparative Analysis of Native iOS Frameworks for Developing Graphics Scenes in Augmented Reality Applications

Paulius Sakalys
Faculty of Electronics and Informatics
Vilniaus kolegija
Vilnius, Lithuania
p.sakalys@eif.viko.lt

Mariya Shirokopetleva
Department of Software Engineering
Kharkiv National University of RE
Kharkiv, Ukraine,
marija.shirokopetleva@nure.ua

Olena Shevchenko
Department of Software Engineering
Kharkiv National University of RE
Kharkiv, Ukraine,
olena.shevchenko@nure.ua

Daria Pervieieva
Department of Software Engineering
Kharkiv National University of RE
Kharkiv, Ukraine,
daria.koshkina@nure.ua

Loreta Savulioniene
Faculty of Electronics and Informatics
Vilniaus kolegija
Vilnius, Lithuania
l.savulioniene@viko.lt

***Abstract*—While defining the technology stack for creating native AR apps for iOS, developers are faced with the need to choose between two popular Apple graphics frameworks: SceneKit and RealityKit. Most available studies are dedicated to the analysis of their internal architecture and most beneficial technical capabilities. Whereas this study focuses on the fundamental comparison of the frameworks using the multi-criteria decision analysis method. For this, the significant qualitative and quantitative comparison criteria are highlighted, as well as a series of experiments are conducted to compare frameworks in terms of scene rendering performance (FPS) and mobile device resource usage (CPU, GPU). Such a formal approach makes it possible to evaluate the effectiveness of frameworks and their suitability for different project types and development teams. This allows developers to make informed decisions during the design stage according to their needs. Confirming the potential of the proposed approach in practice, an example of its application to select the most appropriate framework for two different AR projects is presented. The specificity of applying this approach in the context of the current task was highlighted.**

Keywords— Augmented Reality, graphics scenes, iOS, multi-criteria decision analysis.

I. Introduction

Augmented reality (AR) technologies are becoming an integral part of everyday life, with applications spanning various fields. From mobile gaming to pre-project visualisation and interactive learning, AR enhances the user experience by integrating virtual elements into the real world [1], [2], [3]. According to ARTillary Intelligence [4] in 2024, the number of AR-ready Apple devices surpassed the number of similar devices from Google.

With interactive content becoming a norm, understanding AR frameworks is a crucial skill for software engineers mastering AR. However, integrating AR into iOS applications poses a challenge due to the variety of available frameworks, each with distinct advantages.

Choosing a technology stack when starting a new project is crucial for its development cost, set of features, further updates potential, future success in general and resulting profit. Developers can decide based on their experience and intuition or utilize any suitable formal method. Employing formal analysis can provide a defensible, evidence-based justification of the choice, ensuring that the decision is robust and balanced.

For Apple devices, there are three most commonly used native frameworks that could be utilized to implement high-performance AR applications: ARKit, SceneKit (SK), and RealityKit (RK).

The primary function of ARKit is environmental scanning and real-world integration [5], [6]: detect surfaces, track device position in space, and determine depth for placing virtual objects within the real

Рисунок Б.1 – Перша сторінка тез конференції (рисунок виконаний самостійно)

environment. ARKit can work with LiDAR scanners available on newer iOS devices, allowing the use of three-dimensional spatial data for real-time depth mapping [7]. There is no way to get access to all this functionality without ARKit for Apple devices. ARKit relies on SK, RK, or other graphics engines to render a graphical scene in an AR application [8].

SK is the oldest Apple implementation of rendering technologies. It was introduced in 2012 and was conceived for VR [9]. It is a flexible and powerful 3D rendering engine that enables developers to easily create 3D objects, apply materials, and set up lighting, cameras, animations, and physics simulations, as highlighted in [8], [10].

RK is a modern framework introduced by Apple in 2019 [11]. It enhances 3D object rendering realism in real-world, physics simulations, and spatial audio "out of the box", supports async loading and provides a networking layer to implement state synchronization, etc., as discussed in [10], [12].

If developers want to use only native tools to ensure productivity, they need to choose between SK and RK.

II. MATERIALS AND METHODS

A. Aim and tasks of the study

Despite numerous studies on SK, and RK, most focus is made on analysing their technical capabilities, performance, and integration. However, there are no clear recommendations on selecting the appropriate framework based on project requirements, particularly considering mobile device limitations, performance demands, and rendering specifics. This lack of guidance makes it challenging for developers working on AR applications, highlighting the need for this research.

The aim of this study is to investigate the impact of SK and RK frameworks on the performance and functionality of developed scenes in AR applications for iOS. The research objectives are:

a) Highlight the criteria that can influence the choice of a framework by the software architect, define types of criteria (qualitative or quantitative), measurement scales, ranges, and their possible aggregation groups; the grouping of some criteria is used to reduce their quantity in case it doesn't confuse experts during the criteria groups' weights adjustment.

b) Conduct an experiment to measure criteria values for every alternative (SK and RK frameworks in the current case).

c) Give an example of selected criteria usage to choose SK or RK for specific project and analyse the results.

B. Selecting a formal method for framework scoring

Among the suitable formal methods of alternative selection, there are, for example, neural networks [13], the classical method of multicriteria analysis (MCDA) [14], or its modification based on fuzzy logic [15]. On the one hand, neural networks require a large amount of historical data. This is unlikely to be available to an ordinary software development company. On the other hand, the use of fuzzy logic makes sense when there is partial uncertainty about the values of the selected criteria, or when several experts disagree in their assessments. For such a task as choosing one of two frameworks according to the needs of a particular project, in case all the parameters (criteria) can be measured in terms of certain numerical values and there are no major differences in the way they are interpreted by experts, a classic MCDA is faster and more transparent. This approach also helps to better analyse the pros and cons of the alternatives and to come up with confidence-aware decisions.

For this research classical additive MCDA has been chosen.

C. Criteria identification and definition

There are several aspects that should be considered while making a choice among different graphical frameworks for AR applications. They are performance, supported features (graphics, physics), integration level of AR operations, fine-tuning abilities, simplicity of use, and limitations. Fine-tuning (or customization) ability is the ability to implement such a feature that is not supported in the framework by default.

The values of all indicators are scaled to the selected universal interval scale [0-10], where 0 means not supported or absence of feature and 10 means the best possible level of support. For such parameters as, for example, "Necessity of Performance Tuning," the values are inverted to conform to a unified logic "the bigger - the better".

The performance of AR applications is measured using several key metrics [16]. They are listed in Table I. All the metric ranges in Table I are given non-normalized because it's easier for a person to perceive them in such a format.

The primary performance metric is Frames Per Second (FPS). A high FPS ensures a seamless user experience, whereas a low FPS can cause animation stuttering and degrade usability. For AR\VR applications, the optimal FPS is a stable 60 FPS. Measuring FPS helps assess how well frameworks handle scene rendering while maintaining high performance during animations [17].

CPU usage is critical for overall application performance. It is responsible for loading assets, managing and positioning 3D objects, transferring data to GPU, processing physical interactions and application logic, integrating data from sensors, etc.

Efficient CPU usage is necessary to maintain a balance between animation smoothness and minimal resource consumption [18].

Рисунок Б.2 – Друга сторінка тез конференції (рисунок виконаний самостійно)

TABLE I. PERFORMANCE INDICATORS

Indicator	Range	Description
Average FPS, FPS	0-60	Frames Per Second, indicate the smoothness of the application performance
Occasional FPS Drops, FPS	0-60	FPS drops cause a poor user experience
Initial FPS Drops, FPS	0-60	Min observed FPS drops on pick loads
Max CPU Frequency, GHz	1-3.2	To what max frequency the processor was 'overclocked' at peak loads; this indirectly indicates how much the CPU has been utilized
Max GPU Usage, %	0-100	Shows how much GPU was utilized at pick loads

The lower boundary for CPU Frequency is set to 1 GHz since that's the value for its 'idle' mode. The upper boundary for CPU Frequency is set to 3.2 GHz since for most iPhones (the device on which the experiment was performed) with an AX chip, the maximum frequency of 'large' cores can be approximately in the range of 2.3-3.2 GHz.

GPU utilization plays a key role in achieving high-quality graphics and real-time interaction in AR/VR applications. The GPU is specialized for rendering computations, making it responsible for displaying 3D objects, textures, shadows, reflections, and lighting. In AR applications, the GPU enables virtual objects to blend seamlessly with the real-world camera feed. High computational loads can cause FPS drops and a poor user experience, especially in complex scenes with many objects. To maintain 60 FPS or higher, GPU performance must be optimized to prevent frame rate delays, interface lag, and excessive battery consumption. Overloading the GPU can lead to overheating and rapid battery depletion, making balanced resource management essential for efficient AR application performance [19].

Ensuring framework adaptability allows developers to optimize resource usage while considering device limitations. Effective CPU and GPU management prevents excessive performance degradation and enhances the overall user experience [20].

All the indicators listed in Table I are estimated during the experiment that is described in section II.D as well as the values themselves.

The important functional graphics features are listed in Table II. In addition to the primary criteria, the analysis also considers additional graphics functional aspects [9], [10], [11] that might be crucial to the quality of AR application being developed. They are 3D file formats support (for assets), different animation types (action based, key frame, skeletal, morph target, inverse kinematics, animation blending, etc.), photorealistic quality of the scene (PBR, basic lighting and shadowing, environmental lighting, ray tracing), physical simulations support (collisions, gravity, forces, physical properties of the objects, etc.).

TABLE II. GRAPHICAL FEATURES INDICATORS

Indicator	SK	RK	Description
3D File Formats (GLTF, OBJ, FBX, USDZ, USD)	8	8	SK doesn't support the USD format; RK doesn't support the FBX format
Animations	8	7	Among all animations, SK doesn't support Real-Time AR-Driven animation only while RK doesn't support Inverse Kinematics and Animation State Machines at all or has limited support for Custom Animation Curves, etc.
Photorealism	7	8	For SK is possible and achieves good results with enough tweaking, but it requires more manual work. For RK automatic environmental lighting and reflection systems help deliver a realistic blend with the real world "out of the box".
Physical Simulations	7	6	SK supports common highly customizable physics interactions (rigid bodies, fields, constraints) but is not aimed at advanced fluid/soft-body out of the box. RK is streamlined for typical AR collisions and simple rigid body dynamics but has fewer advanced features or deep customization.
Average score	**7.5**	**7.25**	

Since the target type of applications considered in this paper are AR-oriented applications, the level of AR operations integration into the frameworks being evaluated impacts crucially their development process and, thus, their comparison. For example, handling collision with real-world objects, occlusion (hiding partly or completely virtual objects behind real-world objects), and external real-world light consideration.

According to this indicator, SK has got 6 points while RK has got 9 points. This is due to the fact that SK supports only basic AR Anchoring and Environmental Lighting out of the box. Other functionalities like Real-Time Plane Detection, Real-Time Occlusion, Physics-Based Real-World Interaction, Motion Capture Support, etc. are supported via manual setup with ARKit or not supported at all [10].

RK points are also not maximum because object recognition works based on pre-scanned 3D object models, it also relies heavily on ARKit's spatial awareness capabilities (e.g., tracking, plane detection, and occlusion) [11]. That's why this functionality in RK is significantly weaker than the same functionality in AI-powered frameworks, such as Vuforia and Unity [21]. Manual implementation of such AI-based algorithms is possible with the use of different object

TABLE I. PERFORMANCE INDICATORS

Indicator	Range	Description
Average FPS, FPS	0-60	Frames Per Second, indicate the smoothness of the application performance
Occasional FPS Drops, FPS	0-60	FPS drops cause a poor user experience
Initial FPS Drops, FPS	0-60	Min observed FPS drops on pick loads
Max CPU Frequency, GHz	1-3.2	To what max frequency the processor was 'overclocked' at peak loads; this indirectly indicates how much the CPU has been utilized
Max GPU Usage, %	0-100	Shows how much GPU was utilized at pick loads

The lower boundary for CPU Frequency is set to 1 GHz since that's the value for its 'idle' mode. The upper boundary for CPU Frequency is set to 3.2 GHz since for most iPhones (the device on which the experiment was performed) with an AX chip, the maximum frequency of 'large' cores can be approximately in the range of 2.3-3.2 GHz.

GPU utilization plays a key role in achieving high-quality graphics and real-time interaction in AR/VR applications. The GPU is specialized for rendering computations, making it responsible for displaying 3D objects, textures, shadows, reflections, and lighting. In AR applications, the GPU enables virtual objects to blend seamlessly with the real-world camera feed. High computational loads can cause FPS drops and a poor user experience, especially in complex scenes with many objects. To maintain 60 FPS or higher, GPU performance must be optimized to prevent frame rate delays, interface lag, and excessive battery consumption. Overloading the GPU can lead to overheating and rapid battery depletion, making balanced resource management essential for efficient AR application performance [19].

Ensuring framework adaptability allows developers to optimize resource usage while considering device limitations. Effective CPU and GPU management prevents excessive performance degradation and enhances the overall user experience [20].

All the indicators listed in Table I are estimated during the experiment that is described in section II.D as well as the values themselves.

The important functional graphics features are listed in Table II. In addition to the primary criteria, the analysis also considers additional graphics functional aspects [9], [10], [11] that might be crucial to the quality of AR application being developed. They are 3D file formats support (for assets), different animation types (action based, key frame, skeletal, morph target, inverse kinematics, animation blending, etc.), photorealistic quality of the scene (PBR, basic lighting and shadowing, environmental lighting, ray tracing), physical simulations support (collisions, gravity, forces, physical properties of the objects, etc.).

TABLE II. GRAPHICAL FEATURES INDICATORS

Indicator	SK	RK	Description
3D File Formats (GLTF, OBJ, FBX, USDZ, USD)	8	8	SK doesn't support the USD format; RK doesn't support the FBX format
Animations	8	7	Among all animations, SK doesn't support Real-Time AR-Driven animation only while RK doesn't support Inverse Kinematics and Animation State Machines at all or has limited support for Custom Animation Curves, etc.
Photorealism	7	8	For SK is possible and achieves good results with enough tweaking, but it requires more manual work. For RK automatic environmental lighting and reflection systems help deliver a realistic blend with the real world "out of the box".
Physical Simulations	7	6	SK supports common highly customizable physics interactions (rigid bodies, fields, constraints) but is not aimed at advanced fluid/soft-body out of the box. RK is streamlined for typical AR collisions and simple rigid body dynamics but has fewer advanced features or deep customization.
Average score	**7.5**	**7.25**	

Since the target type of applications considered in this paper are AR-oriented applications, the level of AR operations integration into the frameworks being evaluated impacts crucially their development process and, thus, their comparison. For example, handling collision with real-world objects, occlusion (hiding partly or completely virtual objects behind real-world objects), and external real-world light consideration.

According to this indicator, SK has got 6 points while RK has got 9 points. This is due to the fact that SK supports only basic AR Anchoring and Environmental Lighting out of the box. Other functionalities like Real-Time Plane Detection, Real-Time Occlusion, Physics-Based Real-World Interaction, Motion Capture Support, etc. are supported via manual setup with ARKit or not supported at all [10].

RK points are also not maximum because object recognition works based on pre-scanned 3D object models, it also relies heavily on ARKit's spatial awareness capabilities (e.g., tracking, plane detection, and occlusion) [11]. That's why this functionality in RK is significantly weaker than the same functionality in AI-powered frameworks, such as Vuforia and Unity [21]. Manual implementation of such AI-based algorithms is possible with the use of different object

Рисунок Б.3 – Третя сторінка тез конференції (рисунок виконаний самостійно)

boundary segmentation algorithms, for example, those described in [22].

If the desired feature is not supported by default in the frameworks, it could be possible to implement it manually. Differences between SK and RK in the abilities of feature customization and performance tuning are listed in Table III.

TABLE III. FINE-TUNING INDICATORS

Indicator	SK	RK	Description
Performance Tuning	8	5	SK supports Level of Detail (LOD) management, profiling and reducing draw calls, and simplifying node hierarchies using Xcode Instruments; RK allows to structure "entity-component-system" (ECS) data efficiently
Custo-mization	9	5	SK supports custom shaders, PBR materials, lighting, and advanced postprocessing, e.g. [23], etc. RK supports *only* simple custom materials, reflections, and lighting adjustments
Average score	**8.5**	**5**	

The indicators collected in Table IV [24] are useful because in case other aspects are equal, a product that is easier to use is likely to be preferred.

TABLE IV. USAGE SIMPLICITY INDICATORS

Indicator	SK	RK	Description
Community, Docs, Samples	9	8	SK was introduced in 2012 and RK – in 2019. Both have detailed documentation and a lot of code samples.
Graphical Scene Editor	8	7	SK has an advanced scene editor in Xcode good for deep customization and control over every aspect of a 3D scene but lacks AR-elements integration out of the box; RK uses an external Reality Composer for quick prototyping of AR apps
Scene Organization approach	8	10	SK uses an old-style node-based hierarchy, good for 3D scenes; RK provides a modern and easy-to-start ECS approach, but fewer "classic" 3D scene graph control
Necessity of Performance Tuning	5	9	SK is capable to handle moderately complex scenes with advanced graphical effects but requires manual tuning for performance optimization RK provides a generally efficient graphical engine based on Metal API
Average score	**7.5**	**8.5**	

Limitations for rated frameworks are shown in Table V.

TABLE V. LIMITATIONS INDICATORS

Indicator	SK	RK	Description
Compatibility	10	9	SK is available for all versions of iOS/macOS/visionOS; RK is available for iOS 13+/ macOS 10.15+/visionOS all versions
Scene Complexity Scalability	7	9	SK handles moderately complex scenes, but can struggle without fine performance optimizations; RK is generally efficient for complex scenes due to Metal utilization
Average score	8.5	9	

Platform compatibility influences the potential number of application clients, and Scene scalability affects the classes of AR applications, that could be developed with the corresponding framework.

D. Experiment

To measure the performance indicators of the SK and RK in terms of parameters listed in Table I the experiment was conducted. The ARPerfTester iOS application was developed. The application allows a user to create a three-dimensional scene, add and remove objects, perform animations and analyse system behaviour under varying loads.

Several test scenarios are offered to evaluate the performance of SK, and RK frameworks. The main goal of the testing is to determine how these tools handle rendering a large number of objects, processing animations, user interactions, and utilizing the device's hardware resources.

Test scenario 1: evaluates performance with an increasing number of objects (up to 100, up to 10000 faces in total). The aim is to determine how object count affects rendering performance.

Test scenario 2: test the impact of animations on the performance. The goal is to determine how the framework processes the simultaneous animation of multiple objects. Using a scene with objects (up to 100, up to 10000 faces in total), a simultaneous rotation animation was executed.

Test scenario 3: stability after object removal. The goal is to check system performance as well as memory management efficiency and resource deallocation.

For the experiment, Xcode Instruments [20] were used as the primary tool for performance analysis and debugging of iOS applications. Instruments allow real-time monitoring of device resource usage, including CPU, GPU, FPS, memory, and power consumption, providing precise data on AR application performance.

Рисунок Б.4 – Четверта сторінка тез конференції (рисунок виконаний самостійно)

Before running the tests, the following tools were configured in Xcode Instruments:

- Time Profiler – to analyse CPU load and identify peak processing moments during scene rendering.
- GPU Profiler – to monitor GPU usage while processing 3D objects and animations.
- Core Animation – to track frame rate (FPS) in real-time and assess the stability of the frameworks.

The experiment was conducted on a physical device (iPhone 8 Plus running iOS 16.7.10). During each test scenario, Xcode Instruments recorded all performance metrics, allowing for both real-time analysis and post-test review. The collected data was exported in .trace format, enabling a detailed further analysis of the indicators. The results of the experiment are presented in Table VI below.

TABLE VI. SUMMARY OF THE PERFORMANCE RESULTS

Metric	SceneKit	RealityKit
Average FPS, FPS	58	54
Occasional FPS Drops, FPS	4	9
Initial FPS Drops, FPS	55	9
Max CPU Frequency, GHz	1.68	2.13
Max GPU Usage, %	18	53

Throughout most of the testing, SK indicates high performance. However, short-term drops were observed while adding a large number of objects. These drops occurred mainly in the initial step of changing stages when the system processed a large number of new objects or started to execute simultaneous animations.

RK exhibited more pronounced FPS fluctuations initially (when the stage was still empty) – frame rates dropped to 9 FPS, likely due to the high computational requirements for processing initial physics effects and rendering. Subsequently, FPS stabilized at 57-60 FPS, but periodic performance drops were still recorded. These drops were not as low as for SK, but RK was stabilizing longer than SK. This effect confirms higher system resource demands.

The GPU load in the SK remained relatively low. This suggests efficient computation optimization, where the load is evenly distributed among processes.

RK showed significantly higher GPU usage. The highest peak values were observed during active object interactions when complex lighting, physics effects, and animations were being rendered.

High CPU frequency on SK was observed during the addition of new objects and animation execution, but performance remained stable after object removal. RK indicates significantly higher computational requirements compared to SK. This is due to RK's extensive use of physics simulations and lighting rendering, which increases the overall processor load.

In order to utilize the obtained values for analysis using additive convolution [14], the parameters were normalized to the selected range [0-10] according to (1):

$$x_{norm} = \frac{x_i - x_{min}}{x_{max} - x_{min}} \times 10 \quad (1)$$

The results of the calculations are presented in Table VII.

TABLE VII. NORMALIZED SUMMARY OF THE PERFORMANCE RESULTS

Metric	SceneKit	RealityKit
Average FPS, FPS	9.7	9
Occasional FPS Drops, FPS	0.67	1.5
Initial FPS Drops, FPS	9.17	1.5
Max CPU Frequency, GHz	3.09	5.14
Max GPU Usage, %	1.8	5.3
Average score	**4.89**	**4.49**

E. Application of MCDA to determine the feasibility of frameworks' usage for different applications

Let's describe two typical applications under development that can be used to assess the feasibility of frameworks' usage for their implementation.

Application1 is being developed by a small startup team. Application1 is a product catalogue of furniture and the utility for placing 3D models in space.

Application2 is being developed by a team with experience in 3D graphics or game development. Application2 is the Educational Physics Lab that allows students to place virtual setups (pendulums, mechanical models, etc.) into the real-world environment to investigate and experiment with precise physical interactions, adjusting physical characteristics of the setup components (e.g. motor torque, friction at joints, gear ratios, angular constraints, mass distribution, damping, joint strength, etc.).

Through the analysis of application functionality, weight values for each group of indicators have been proposed for each of the applications. The results of additive convolution [14] for Application1 for both frameworks are given in Table VIII, and for Application 2 are given in Table IX.

TABLE VIII. RESULTS FOR APPLICATION1

Criterion	Weight	SceneKit		RealityKit	
		c_i	$c_i * w_i$	c_i	$c_i * w_i$
Performance	0,25	4.89	1.22	4.49	1.12
Graphical Features	0,1	7.5	0.75	7.25	0.73
AR Integration	0,25	6	1.5	9	2.25
Fine-Tuning	0,1	8.5	0.85	5	0.5
Simplicity	0,2	7.5	1.5	8.5	1.7
Limitations	0,1	8.5	0.85	9	0.9
Total			6.67		**7.2**

Рисунок Б.5 – П'ята сторінка тез конференції (рисунок виконаний самостійно)

TABLE IX. RESULTS FOR APPLICATION2

Criterion	Weight	SceneKit		RealityKit	
		c_i	$c_i * w_i$	c_i	$c_i * w_i$
Performance	0,25	4.89	1.22	4.49	1.12
Graphical Features	0,3	7.5	2.25	7.25	2.18
AR Integration	0,15	6	0.9	9	1.35
Fine-Tuning	0,2	8.5	1.7	5	1
Simplicity	0,05	7.5	0.38	8.5	0.43
Limitations	0,05	8.5	0.43	9	0.45
Total			**6.88**		6.53

The experiment demonstrates that RK is more suitable for Application1, whereas SK is preferable for Application2.

This is because for Application1 basic photorealism and support for various AR interactions, such as gestures to move and rotate objects, are fundamental, while Application2 is impossible without accurate modeling of complex physical interactions and careful performance tuning for complex graphical objects.

III. Results and Discussion

The obtained results demonstrate that the SK is more stable in maintaining frame rates and efficiently utilizes hardware resources, particularly the GPU, thus is suitable for projects where performance and energy optimization are critical. Conversely, with the same scene project in code RK provides more photorealistic rendering and complex physics interaction, but this comes at the cost of significantly higher CPU and GPU load, which leads to FPS instability and increased power consumption. This, in general, confirms the developers' observations obtained in [10], but Apple is constantly improving RK and Metal that is used in its background (instead of OpenGL, which is used by default in SK) and, in perspective, RK should outperform the SK.

According to the features and capabilities comparison, both frameworks cover the same set of them, but there are differences in specific features and how many of them are built-in versus needed manual work. In general, SK is more powerful, mature and flexible in graphical, physical and audio features, but it needs more manual work and professional knowledge to develop them. RK turns on, by default, simple variants of these features but is not able to implement extended versions of them at all. For example, specialized shader effects (non-realistic "Cel" or "toon"-rendering with discrete colour bands, sharp lightning transitions and outlined edges) are not possible in RK at all. They could be useful in educational AR applications, artistic visualization or interactive guides, where it is necessary to help users easily distinguish virtual AR content from real-world scenes.

The same situation is in physical simulations. Powered by Bullet [25], SK allows to set up advanced physical properties of graphical objects, as for Application2. Such interactions are very hard to implement in RK which lacks the granularity needed to provide educationally accurate, highly customizable demonstrations of complex mechanical principles.

On the other side, as the study showed, RK outperforms SK in AR features support. RK was built for AR, so it has AR conveniences that SK alone doesn't (e.g. automatic gesture handling). A crucial weakness of SK is that under visionOS it runs only in 2D mode, lacking visionOS immersive integration at all.

As a more modern development tool, RK is simpler (though less customizable) than SK for less experienced developers. SK suites developers teams who are professionals in 3D graphics.

In general, RK emerges as the recommended choice for new AR-specific projects. Developers should also consider RK to prototype and future-proof their applications, especially in the context of Apple's spatial computing initiatives, including visionOS. Nevertheless, SK continues to serve as a reliable general-purpose 3D graphics framework, particularly useful when AR capabilities are secondary, or there are some features that are much harder or not possible to implement in RK than in SK.

IV. Conclusions

This study conducted a comprehensive analysis of the performance and capabilities of the SK and RK frameworks in the context of developing AR applications for iOS. Key evaluation criteria were identified and assessed. A method of MCDA was offered to rigorously select a framework that better meets the requirements of a specific project. It is necessary to pay attention to the peculiarity of the MCDA application in the case when there are mandatory ("must-have") features, e.g. vertex-shaders control or custom graphical post-processing effects. Such mandatory criteria should be excluded from MCDA and checked first. The following MCDA should be made among the alternatives that have passed the mandatory criteria test.

Regarding the future perspective, it is necessary to focus on the research of AI-driven methods in AR, such as detailed spatial understanding, semantic scene segmentation, and mesh classification [26],[27], which could greatly enhance AR functionality for native iOS applications.

REFERENCES

[1] M. K. Shaleh Md Asari, N. M. Suaib, M. H. Abd Razak, M. A. Ahmad, and N. M. Shaleh, "Empowering Skill-Based Learning with Augmented Reality and Virtual Reality: A Case Study," in 2024 IEEE International Symp. on Consumer Technology, Kuta, Bali, Indonesia, 2024, pp. 225-229, https://doi.org/10.1109/ISCT62336.2024.10791270.

[2] B. Ton, N. Tempert, and D. Plass, "Immersive visualisation of point cloud data of railway environments," in 2024 10th International Conf. on Virtual Reality, Bournemouth, United Kingdom, 2024, pp. 233-239, https://10.1109/ICVR62393.2024.10868576.

Рисунок Б.6– Шоста сторінка тез конференції (рисунок виконаний самостійно)

[3] A. Dhiya' Mardhiyyah, Vincent, K. L. Mario Gracius, F. Permana, and F. I. Maulana, "LonelyScape: Increasing Attractiveness of Escape Room Game Using Augmented Reality Technology," in 2023 International Conf. on Information Management and Technology, Malang, Indonesia, 2023, pp. 795-800, https://doi.org/10.1109/ICIMTech59029.2023.10277954

[4] MobiDev,"12 Augmented Reality Technology Trends of 2025: New Milestones in Immersive Innovations," MobiDev, 2024. [Online]. Available: https://mobidev.biz/blog/augmented-reality-trends-future-ar-technologies. [Accessed: Nov. 17, 2024].

[5] Th. A. R. Sure, "Motion Tracking in iOS Applications Using Augmented Reality," Journal of Android and iOS Applications and Testing,vol. 8, no. 3, pp.1-5, Oct. 2023.

[6] L. Nissen et al., "Towards Preventing Gaps in Health Care Systems Through Smartphone Use: Analysis of ARKit for Accurate Measurement of Facial Distances in Different Angles," Sensors, vol. 23, no. 9, 4486, May 2023, https://doi.org/10.3390/s23094486.

[7] Apple, "Apple Developer Documentation ARKit," 2024. [Online]. Available: https://developer.apple.com/documentation/arkit. [Accessed: Nov. 21, 2024].

[8] J. Nhan, "Building Your First ARKit App with SceneKit," in Mastering ARKit, Berkeley, CA: Apress, 2022, pp. 14-27.

[9] Apple, "Apple Developer Documentation. SceneKit," 2024. [Online]. Available: https://developer.apple.com/documentation/scenekit. [Accessed: Nov. 27, 2024].

[10] "RealityKit vs SceneKit vs Metal – High-Quality Rendering," Mar. 7, 2024. [Online]. Available: https://stackoverflow.com/questions/60505755/realitykit-vs-scenekit-vs-metal-high-quality-rendering. [Accessed: Jan. 17, 2025].

[11] Apple, "Apple Developer Documentation. RealityKit," 2024. [Online]. Available: https://developer.apple.com/documentation/realitykit. [Accessed: Dec. 1, 2024].

[12] S. Sasmoko, F. L. Gaol, and T. Oktavia, "Augmented Reality SDK Overview for General Application Use," International Journal of Advanced Computer Science and Applications, vol. 14, no. 11, pp. 54-60, 2023, http://doi.org/10.14569/IJACSA.2023.0141106.

[13] S. Wang, B. Mo, and J. Zhao "Deep neural networks for choice analysis: Architecture design with alternative-specific utility functions," Transportation Research Part C: Emerging Technologies, vol. 112, pp. 234-251, Mar. 2020, https://doi.org/10.1016/j.trc.2020.01.012.

[14] A. Ishizaka and P. Nemery, Multi-Criteria Decision Analysis: Methods and Software. Hoboken, NY: Wiley, 2013, https://doi.org/10.1002/9781118644898.

[15] L. Maretto, M. Faccio, and D. Battini, "A Multi-Criteria Decision-Making Model Based on Fuzzy Logic and AHP for the Selection of Digital Technologies," IFAC-PapersOnLine, vol. 55, no. 2, pp. 319-324, 2022, https://doi.org/10.1016/j.ifacol.2022.04.213.

[16] P. Nowacki and M. S. Woda "Capabilities of ARCore and ARKit Platforms for AR/VR Applications," Engineering in Dependability of Computer Systems and Networks, vol. 987, pp. 358–370, 2020.

[17] A.M. Boutsi, C. Ioannidis, and S. Verykokou "Multi-Resolution 3D Rendering for High-Performance Web AR," Sensors, vol. 23, no. 15, 6885, Aug. 2023, https://doi.org/10.3390/s23156885.

[18] M. Hort, M. Kechagia, F. Sarro, and M. Harman "A Survey of Performance Optimization for Mobile Applications," IEEE Transactions on Software Engineering, vol. 48, no. 8, pp. 2879–2904, Aug. 2022, https://doi.org/10.1109/TSE.2021.3071193.

[19] F. Nusrat, F. Hassan, H. Zhong, and X. Wang. "How Developers Optimize Virtual Reality Applications: A Study of Optimization Commits in Open Source Unity Projects," in ICSE '21: Proceedings of the 43rd International Conf. on Software Engineering, pp. 473–485, Nov. 2021, https://doi.org/10.1109/ICSE43902.2021.000.

[20] "Getting Started with Instruments. WWDC Notes," 2019. [Online]. Available: https://wwdcnotes.com/documentation/wwdcnotes/wwdc19-411-getting-started-with-instruments. [Accessed: Oct. 17, 2024].

[21] "Vuforia Instruct Is Now Vuforia Expert Capture," May 2023. [Online]. Available: https://www.ptc.com/en/products/vuforia/vuforia-instruct. [Accessed: Jan. 17, 2025]

[22] K. Smelyakov, S. Smelyakov., and A. Chupryna, "Chapter 1. Adaptive Edge Detection Models and Algorithms," in Advances in Spatio-Temporal Segmentation of Visual Data.Studies in Computational Intelligence, vol. 876: Springer, Cham, 2020, pp.1-51, https://doi.org/10.1007/978-3-030-35480-0_1.

[23] K. Smelyakov, M. Hvozdiev, A. Chupryna, D. Sandrkin, and V. Martovytskyi, "Comparative EfficiencyAnalysis of Gradational Correction Models of Highly Lighted Image," in 2019 IEEE International Scientific-Practical Conf. Problems of Infocommunications, Science and Technology, Kyiv, Ukraine, 2019, pp. 703-708, https://doi.org/10.1109/PICST47496.2019.9061356.

[24] I. Gruzdo, I. Kyrychenko, G. Tereshchenko, N. Shanidze, "Metrics Applicable for Evaluating Software at the Design Stage," in COLINS-2021: 5th International Conference on Computational Linguistics and Intelligent Systems, vol. I (2870): Main Conference, Lviv, Ukraine, 2021, pp.916-936. [Online]. Available: https://ceur-ws.org/Vol-2870. [Accessed: Nov. 21, 2024].

[25] C. Dickinson, Learning Game Physics With Bullet Physics and OpenGL, Packt Pub Ltd, 2013.

[26] R. Yenni and A. P V, "Semantic Segmentation and Spatial Relationship Modeling in Hyperspectral Imagery Using Deep Learning and Graph-Based Representations," in 2024 14th Workshop on Hyperspectral Imaging and Signal Processing: Evolution in Remote Sensing, Helsinki, Finland, 2024, pp. 1-4, https://doi.org/10.1109/WHISPERS65427.2024.10876420.

[27] N. Ali, A. Z. Ijaz, R. H. Ali, Z. Ul Abideen and A. Bais, "Scene Parsing Using Fully Convolutional Network for Semantic Segmentation," in 2023 IEEE Canadian Conf. on Electrical and Computer Engineering, Regina, SK, Canada, 2023, pp. 180-185, https://doi.org/10.1109/CCECE58730.2023.10288934.

Рисунок Б.7 – Сьома сторінка тез конференції (рисунок виконаний самостійно)

ДОДАТОК Д

Експертний висновок результатів перевірки кваліфікаційної роботи на відповідність оформлення вимогам ДСТУ 3008: 2015

Експертний висновок результатів перевірки кваліфікаційної роботи

студент (посада) | програмної інженерії (кафедра) | ІПЗм-23-1 (група)

Первєєва Дар'я Олександрівна
(прізвище, ім'я, по батькові)

Зауваження

Пункт ДСТУ 3008-2015	Зміст пункту	Сторінка кваліфікаційної роботи
1	2	3
	7.1 Загальні положення	
	7.3 Нумерація сторінок звіту	
	7.4 Нумерація розділів, підрозділів, пунктів, підпунктів	
	7.5 Рисунки	
	7.6 Таблиці	
	7.7 Переліки	
	7.8 Примітки	
	7.9 Виноски	
	7.10 Формули та рівняння	
	7.11 Посилання	
	7.13 Список авторів	
	7.14 Скорочення та умовні познаки	
	7.15 Додатки	

зауважень немає

Експерт _____ (підпис) Олена ОЛІЙНИК (прізвище, ініціали)

11.06.2025