Лабораторна робота №2

Тема: Створення настільної AR-гри з використанням технології доповненої реальності

Мета роботи: набути практичні навички з використання технології доповненої реальності при створенні ігор

Теоретичні відомості:

Для створення застосунку AR-гри використовується Vuforia SDK, інтегрованого з двигуном Unity. Ключові компоненти:

- Налаштування передачі даних із камери доповненої реальності
- Відстеження зображень – що це і як працює
- Прикріплення цифрового контенту до фізичного об'єкту
- Створення взаємодій, що налаштовуються, що запускаються в залежності від того, що бачить камера

Навчальний курс по Unity можна пройти за посиланням Getting Started In Unity .

За замовчуванням Vuforia працює із *розпізнаванням образів (Image Recognition)* . Розпізнавання образів, також називається розпізнаванням трекерів (Tracker Recognition) або трекінгом зображень (Image Tracking) - це процес, при якому камера розпізнає заздалегідь задане зображення і знає, що з ним робити, наприклад, виконувати поверх нього рендеринг якогось контенту. Найкраще це працює тоді, коли зображення трекінгу якимось чином відповідає контенту, наприклад, плани поверхів для рендерингу будівель добре працюють поверх зображення з будинком.

*Як це працює:*дуже важливо добре підібрати якісне зображення для трекінгу. При використанні Vuforia можна завантажити вибране зображення на портал розробника, щоб перевірити якість його відстеження, і це потрібно робити , *перш* ніж починати розробку. Портал розробника надає зображенню рейтинг, але, що найважливіше, показує його «характерні точки». Щоб трекер був хорошим, ці характерні точки (feature points) повинні бути густо розподілені за зображенням і в них не повинно бути повторюваних патернів. Під час виконання камера шукає ці характерні точки, щоб визначити своє розташування щодо зображення.

В результаті виконання завдання 1отримаємо гру за зразком:

Рис. 1. Приклад застосунку гри з використанням технології доповненої реальності

Завдання на лабораторну роботу

Завдання 1. Створення AR-гри «Випікання піци».

1. Відкрийте Unity, Vuforia та налаштувати веб-камеру.
2. Завантажне приклад проекту:
https://drive.google.com/file/d/1oej_DNfn0rsXg0omXY6ZDMdGQzS6qjPL/view?usp=sharing
3. Після завантаження витягніть файли і відкрийте в Unity проект *How to Make an AR Game Using Vuforia Starter*. Коли проект завантажиться, відкрийте сцену *Starter* із папки *Scenes* та подивіться на вікно Hierarchy:

Рис.2. Вигляд вікна Hierarchy

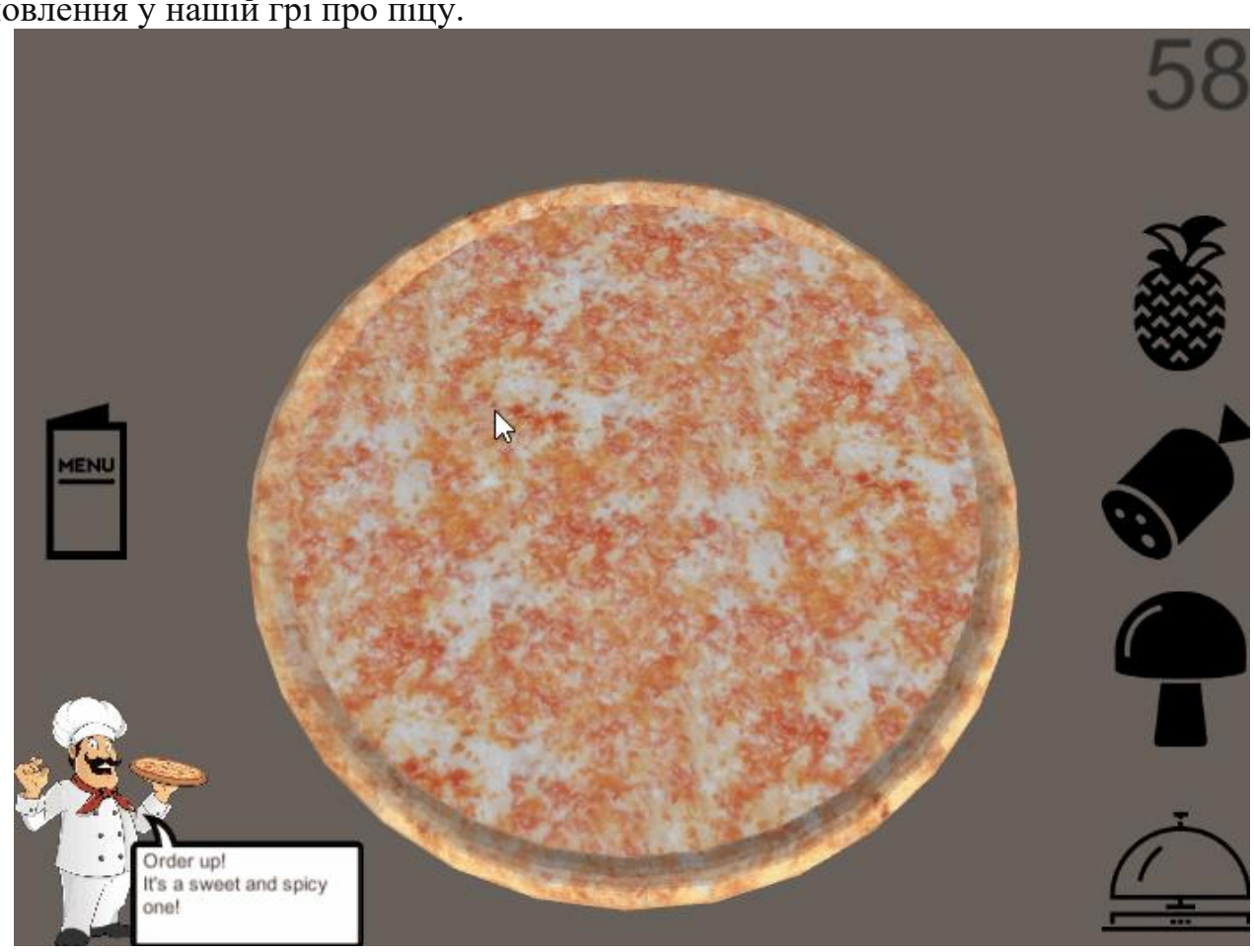
4. Переглянувши склад проекту, натисніть кнопку *Play* у редакторі, щоб заповнити замовлення у нашій грі про піцу.
5. 

Рис. 3. Вигляд гри для замовлення піци

Відкривши додаток, можна пограти з піцою, але наше завдання полягає в тому, щоб перетворити це на AR-гру.

Примітка: асети UI для цього завдання доступні до завантаження з shareicon.net

6. Знайомимося з шеф-кухарем Vuforia

Щоб це зробити, нам потрібно змусити віртуальну піцу відображатися на зображенні-трекері. Після додавання на замовлення начинки потрібно ніби фізично «подати» піцу, винісши її за межі видимості камери.

6.1. Для початку потрібно замінити *Main Camera* на *AR Camera* . Видаліть зі сцени *Main Camera*, а потім *натисніть правою клавішею миші* Hierarchy.

6.2. Додайте *Vuforia -> AR Camera*. При цьому з'явиться це спливаюче вікно:

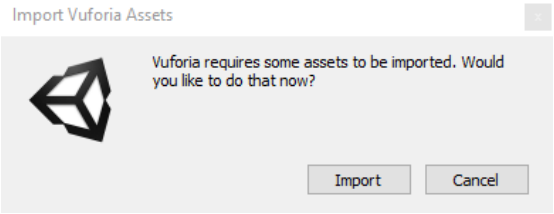


Рис.4. Спливаюче вікно при додаванні AR-камери

6.3. Оберіть *Import* і дочекайтеся, поки Vuforia Package імпортується в проект, додасться досить багато файлів.

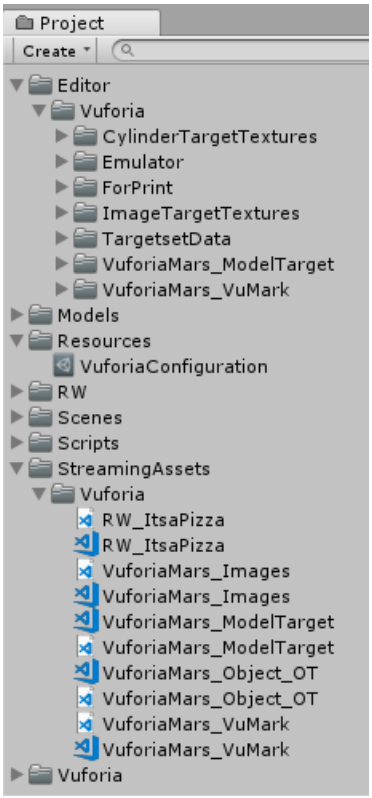


Рис.5. Додавання файлів

Примітка з’являться шаблони додані до проекту. У папці *Vuforia* зберігаються всі префаби та скрипти, що використовуються платформою Vuforia. Варто також згадати про файл *VuforiaConfiguration* всередині папки *Resources* . Цей файл доданий, тому що для розробки програми Vuforia потрібна ліцензія.

*Примітка 2:*якщо у вашому проекті немає ліцензійного ключа програми (App License Key), його можна знайти у файлі README. Скопіюйте рядок у полі інспектора файлу *VuforiaConfiguration* .

6.4. Включіть доповнену реальність:

6.4.1. Перейдіть в *Edit->Project Settings->Player* . Прокрутіть вниз до параметрів *XR Settings* і переконайтеся, що поставлено прапорець *Vuforia Augmented Reality Supported* .

6.4.2. Натисніть у редакторі *Play* та скажіть «привіт» самому собі!

Примітка: якщо ви натиснули на play і з'явився екран "Vuforia Initialization Failed", спробуйте перезапустити Unity. Це випадковий баг, який з'являється лише на деяких системах.

7. Введення у розпізнавання образів

7.1. Для цього завдання зображення та база даних трекінгу вже налаштовані. Зображення піци знаходиться в папці Materials, яку ви завантажили раніше. В ідеалі, варто його роздрукувати. Або ж його можна відкрити на якомусь цифровому пристрої, а потім «показати» його камері. Ось як виглядає зображення у Vuforia Tracker Database:

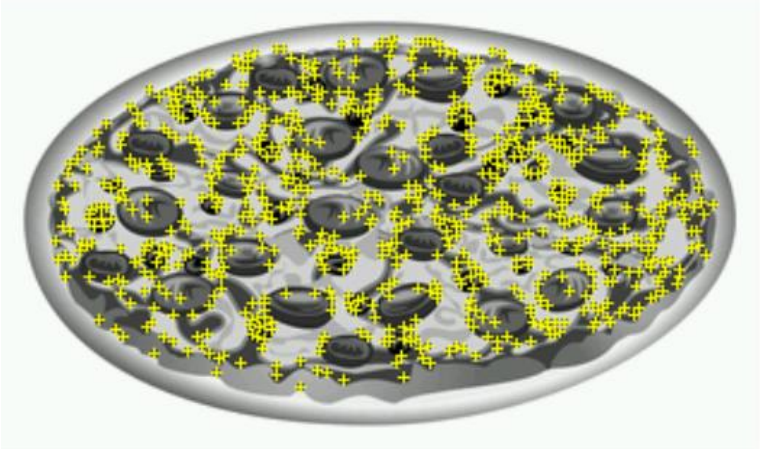


Рис.6. Вигляд зображення піи в Vuforia Tracker Database

7.2. Додавання до сцени Image Targets

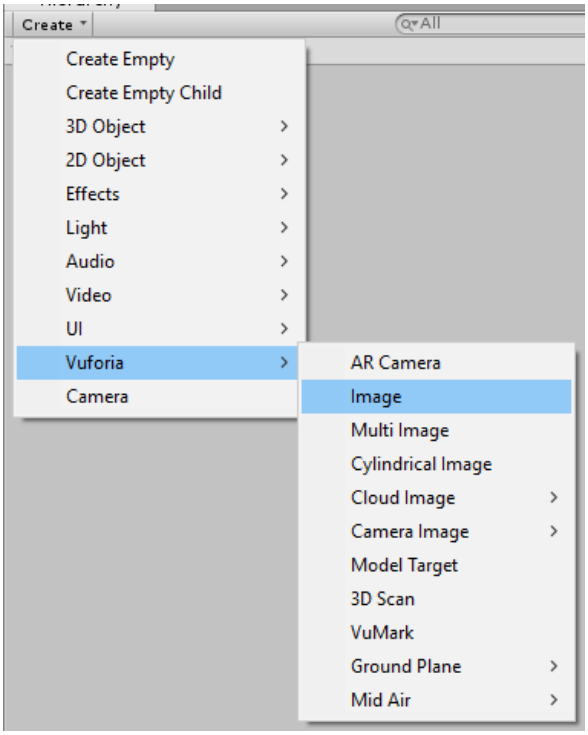


Рис.7. Додавання Image Targets

7.3. Ми зробимо так, щоб поверх піци з'являлася віртуальна піца. Знову скористаємося меню *Create* в *Hierarchy* і виберемо *Vuforia -> Image* .

7.4. Тепер у нас у сцені є *Image Target GameObject* . Подивіться цей target в інспекторі, і ви побачите кілька компонентів. Найважливішими є *Image Target Behavior* та *Default Trackable Event Handler* .

7.5. Для *Database* вибрано значення *RW_ItsaPizza*

7.6. Для *Image Target* вибрано *PizzaClipArt*

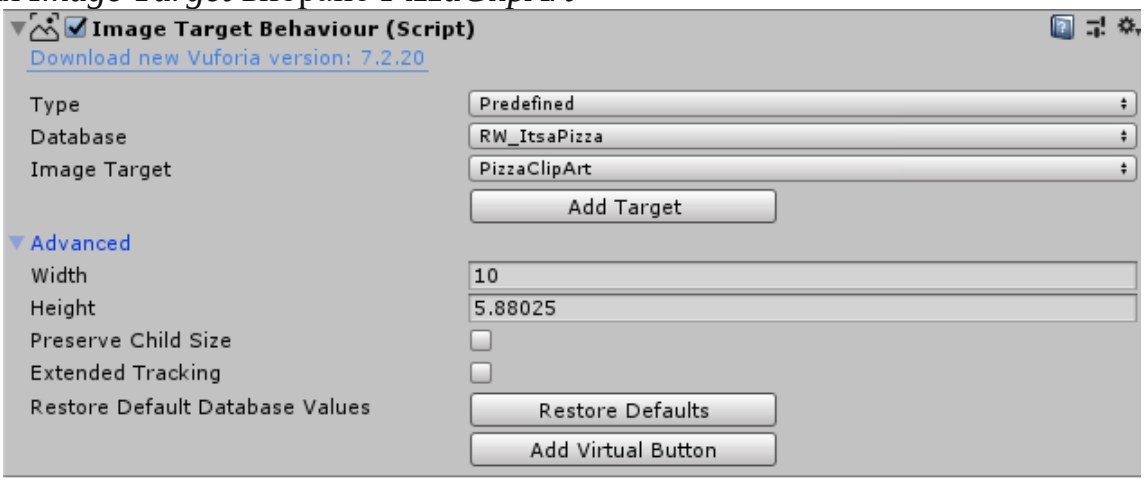


Рис.8. Вибір для *Image Target*

Примітка: трекер Pizza – єдиний image target у вашій базі даних; однак, в одній базі даних можуть бути сотні зображень. Крім того, в одному додатку може бути кілька баз даних.

7.7. Тепер, коли ми налаштували у сцені *AR Camera* та *Image Tracker* , у нас є все необхідне для роботи доповненої реальності! Натисніть Editor на кнопку *Play* і поставте роздруковане зображення навпроти камери.

Рис. 9. В нас з'явилася піца!

8. Прикріплення Game Objects до Trackers як дочірні об'єкти

Піца трохи маленька, але вона міцно приклеїлася до зображення-трекеру. Можна також помітити, що якщо прибрати зображення, що відстежується, то піца залишиться висіти в повітрі.

8.1. Справа в тому, що поки веб-камера може бачити *зображення-трекер* , Vuforia здатний оновлювати позицію *AR Camera* у сцені. Якщо хочете побачити це в дії, налаштуйте Unity Editor таким чином, щоб були одночасно видні вікна *Game* і *Scene* , потім виберіть *AR Camera* і натисніть Play.

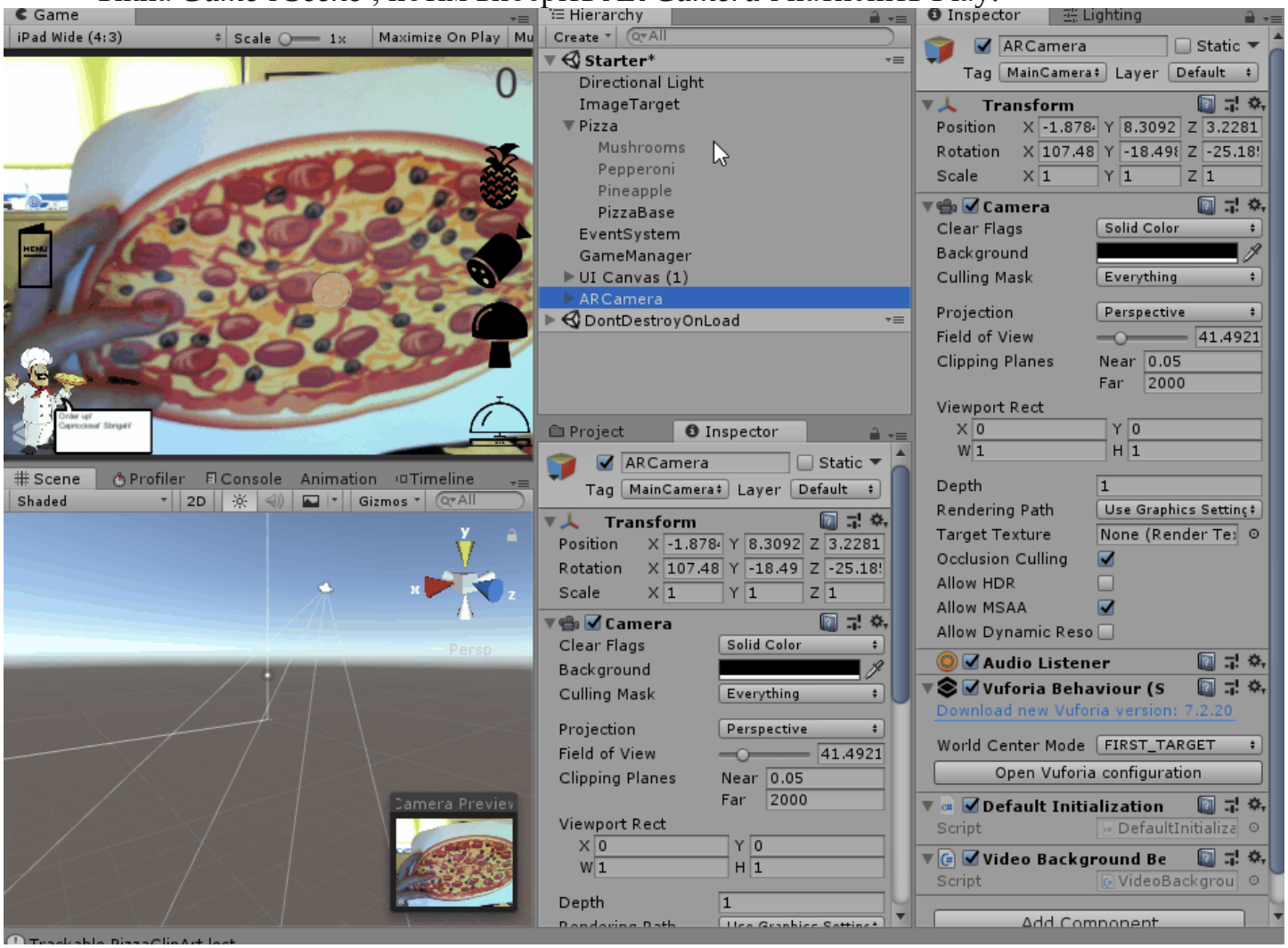


Рис. 10. Приклад оновлення позиції з використанням Vuforia

8.2. Як же змусити піцу поводитися правильно, коли працює камера? Виберіть *ImageTarget* у Hierarchy. Ви побачите, що його масштаб кожної осі має значення *10* . Цим управляє компонент *Image Target Behaviour*. У розділі Advanced можна побачити, що параметр *Width* має значення *10*. Він був заданий під час завантаження зображення на сайт Vuforia Developer.

Підказка: ви не розумієте, чому Image Target у цьому завданні була задана ширина 10? Коли розмір не є суттєвим фактором для програми, то установка значення 10 забезпечує камері підвищену ймовірність трекінгу, зберігаючи при цьому хорошу позицію для контенту між ближньою і дальньою площинами AR Camera. Однак іноді потрібно, щоб доповнена реальність мала певний масштаб. У такому випадку потрібно встановити розмір зображення-трекера таким чином, щоб він відповідав фізичним вимірюванням при завантаженні на портал розробника.

8.3. Тепер виберіть GameObject *Pizza* в *Hierarchy* . Перетягніть його на *ImageTarget* , щоб зробити його дочірнім елементом. Масштаб зміниться на (X: 0.1, Y: 0.1, Z:

0.1). Поверніть його до значень (X: 1, Y: 1, Z: 1), а також перемістіть Position вгору до 0.01осі *Y* . Завдяки цьому *Pizza* буде відповідати *ImageTarget* .

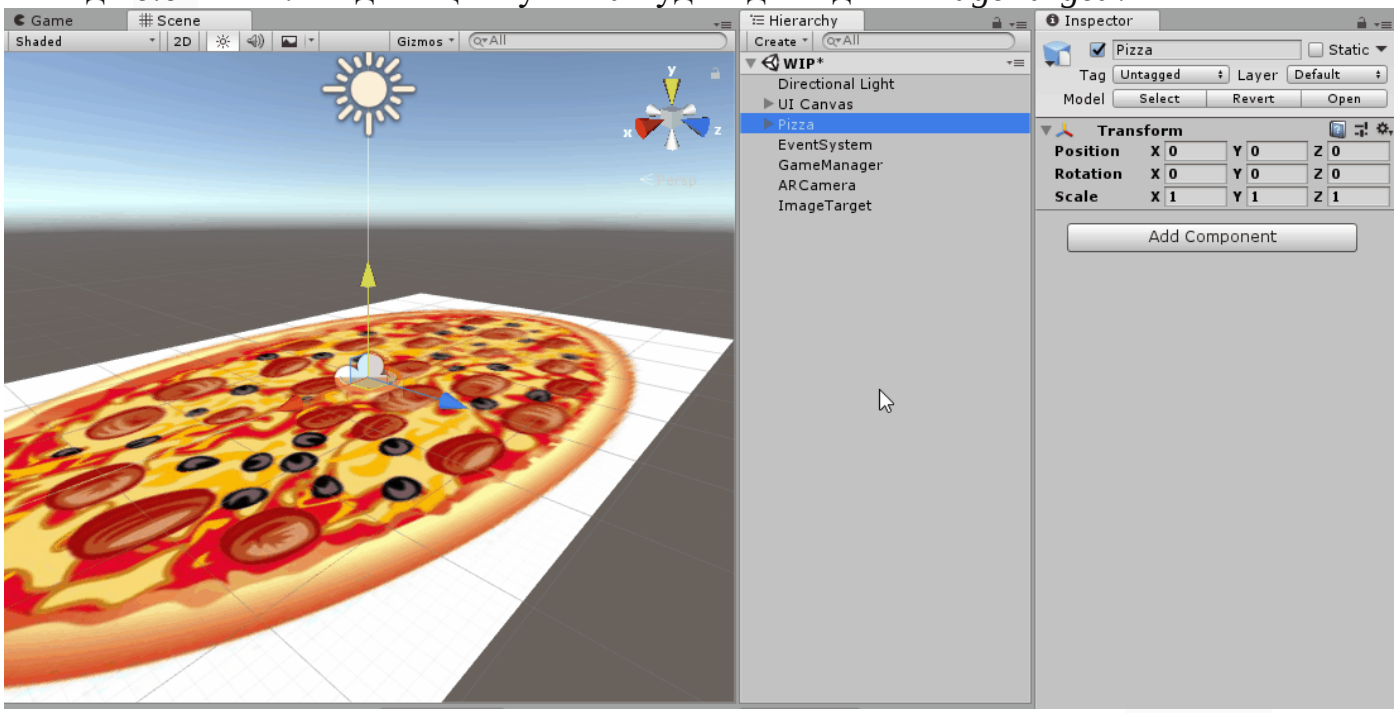


Рис. 11. Оберти по вісям, виставлення відповідних значень по X, Y та Z

8.4. Знову натисніть *Play* і побачите, що піца знаходиться поверх зображення, а також зникає, коли зображення пропадає з кадру.

Рис. 12. Демонстрація знаходження піци поверх зображення в режимі гри

8. Вивчення DefaultTrackableEventHandler

8.1. поведінка береться з *DefaultTrackableEventHandler* об'єкта *ImageTarget* . Відкрийте скрипт і перегляньте його.

Скрипт прокоментований, але варто звернути увагу на деякі аспекти:

- Функція *Start* реєструє цей скрипт як Event Handler (обробник подій) для *TrackableBehaviour* (у разі це *ImageTargetBehaviour*).
- *OnDestroy* видаляє це посилання.
- *OnTrackableStateChanged* - найважливіша функція. Її код повідомляє, що має відбуватися при зміні стану трекінгу.
- *OnTrackingFound* і *OnTrackingLost* викликаються з *OnTrackableStateChanged* . У *DefaultTrackableEventHandler* вони перемикають компоненти *Renderer* , *Collider* та *Canvas* будь-якого дочірнього об'єкта.

Коли камера виявляє зображення, вона не просто переміщає *AR Camera* ; вона також наказує GameObject *Pizza* увімкнути всі його компоненти *Renderer* , а коли зображення пропадає з камери, він знову наказує відключити їх.

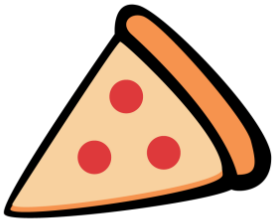


Рис. 13. Зображення піци

9. Створення власних дій трекінгу

9.1 Видаліть із *ImageTarget* компонент *DefaultTrackableEventHandler*. Потім додайте *PizzaTrackableEventHandler*, який можна знайти у папці *Scripts*. Потім відкрийте *PizzaTrackableEventHandler*. Це клон *DefaultTrackableEventHandler*, але код *OnTrackingFound* і *OnTrackingLost* видалений - це завдання вирішіть саамостійно.

9.2 Увімкнення та відключення компонентів *Renderer* знадобиться майже в кожному AR-програмі, тому поверніть цей код. При виникненні проблем, його то можна скопіювати його з *DefaultTrackableEventHandler* або переглянути нижче.

Код

```
protected virtual void OnTrackingFound()
{
    var rendererComponents = GetComponentsInChildren<Renderer>(true);

    // Enable rendering:
    foreach (var component in rendererComponents)
    {
        component.enabled = true;
    }
}

protected virtual void OnTrackingLost()
{
    var rendererComponents = GetComponentsInChildren<Renderer>(true);
```

```
    // Enable rendering:
    foreach (var component in rendererComponents)
    {
        component.enabled = false;
    }
}
```

10. Перетворіть даний проект на гру в доповненій реальності!
11. Подивіться на UI та знайдіть кнопку, яку потрібно натиснути гравцю, щоб завершити свою піцу.

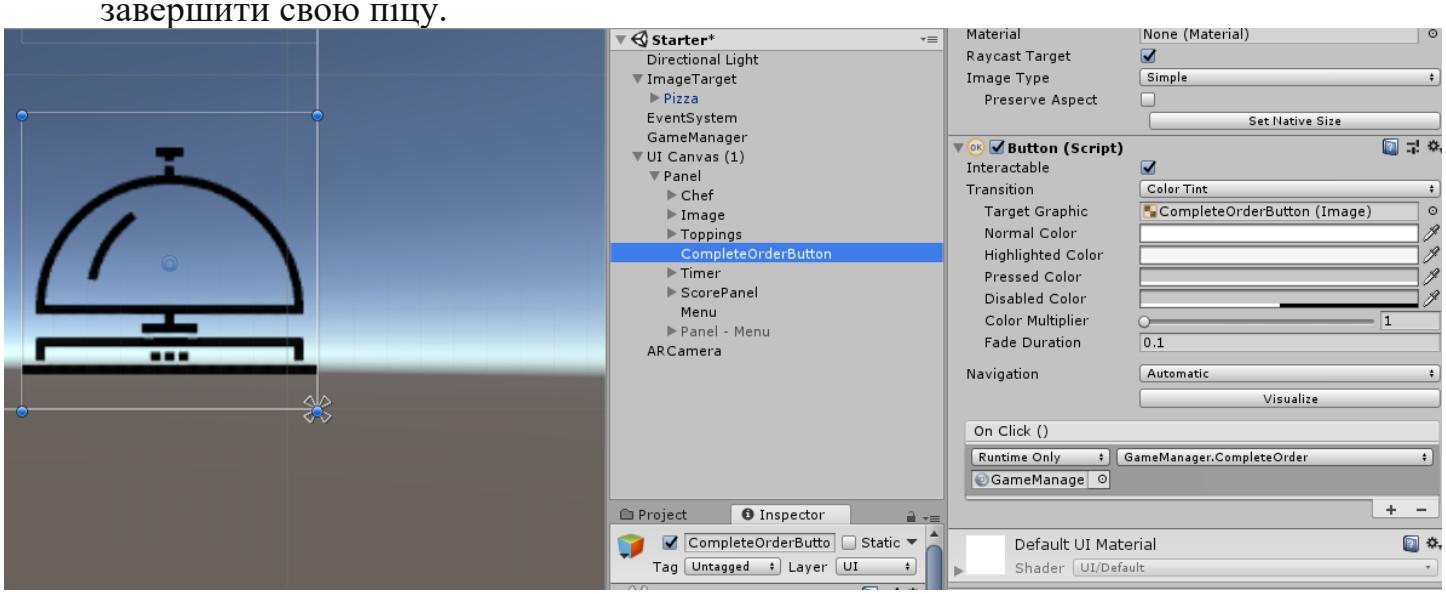


Рис. 14. Кнопка завершення приготування піци

При натисканні на кнопку викликається Event GameManager.CompleteOrder().

12. Замість того, щоб змушувати гравця натискати кнопку, зробіть так, щоб для завершення замовлення він «подавав» піцу (переміщував трекер за межі видимості камери).
13. На початку *PizzaTrackableEventHandler* додамо *UnityEvent* для виклику, коли Image втрачає трекінг.

```
using Vuforia;
using UnityEngine;
using UnityEngine.Events;

public class PizzaTrackableEventHandler : MonoBehaviour, ITrackableEventHandler
{
    public UnityEvent OnTrackingLostEvent;
...
```

14. Опишіть виклик події у методі *OnTrackingLost* :

```
protected virtual void OnTrackingLost()
{
    var rendererComponents = GetComponentsInChildren<Renderer>(true);
```

```
    // Enable rendering:
    foreach (var component in rendererComponents)
    {
        component.enabled = false;
    }
    //Trigger our event
    OnTrackingLostEvent.Invoke();
}
```

Даний код додає *PizzaTrackableEventHandler* більшої гнучкості, тому що тепер можна задавати виконання будь-яких дій у разі втрати трекінгу.

15. Збережіть *PizzaTrackableEventHandler* , поверніться до Unity Editor і дочекайтеся завершення компіляції.
16. Після завершення компіляції, зробіть так, щоб GameManager.CompleteOrder() викликався при втраті трекінгу піци.
17. Вимкніть або видаліть *CompleteOrderButton* в UI.

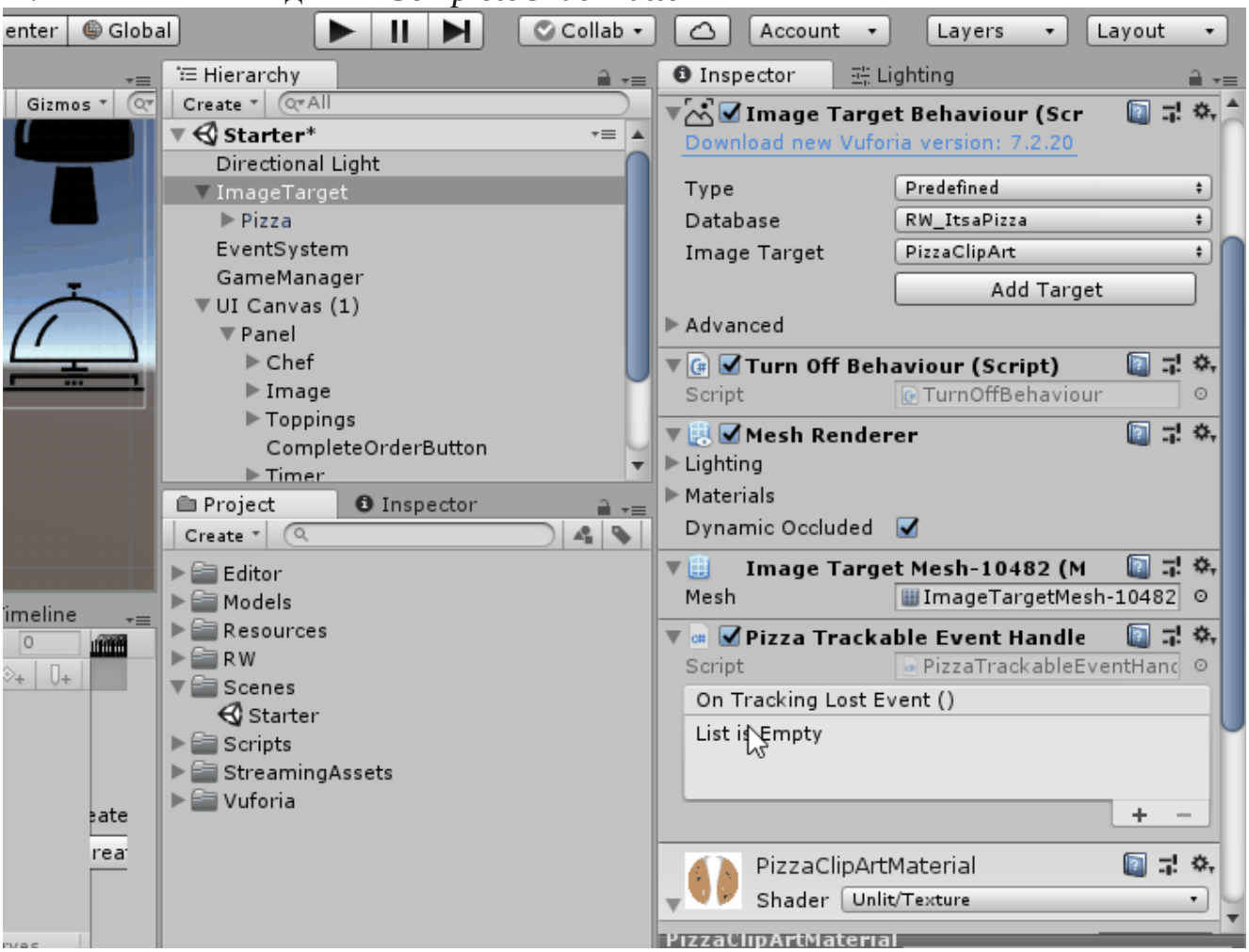
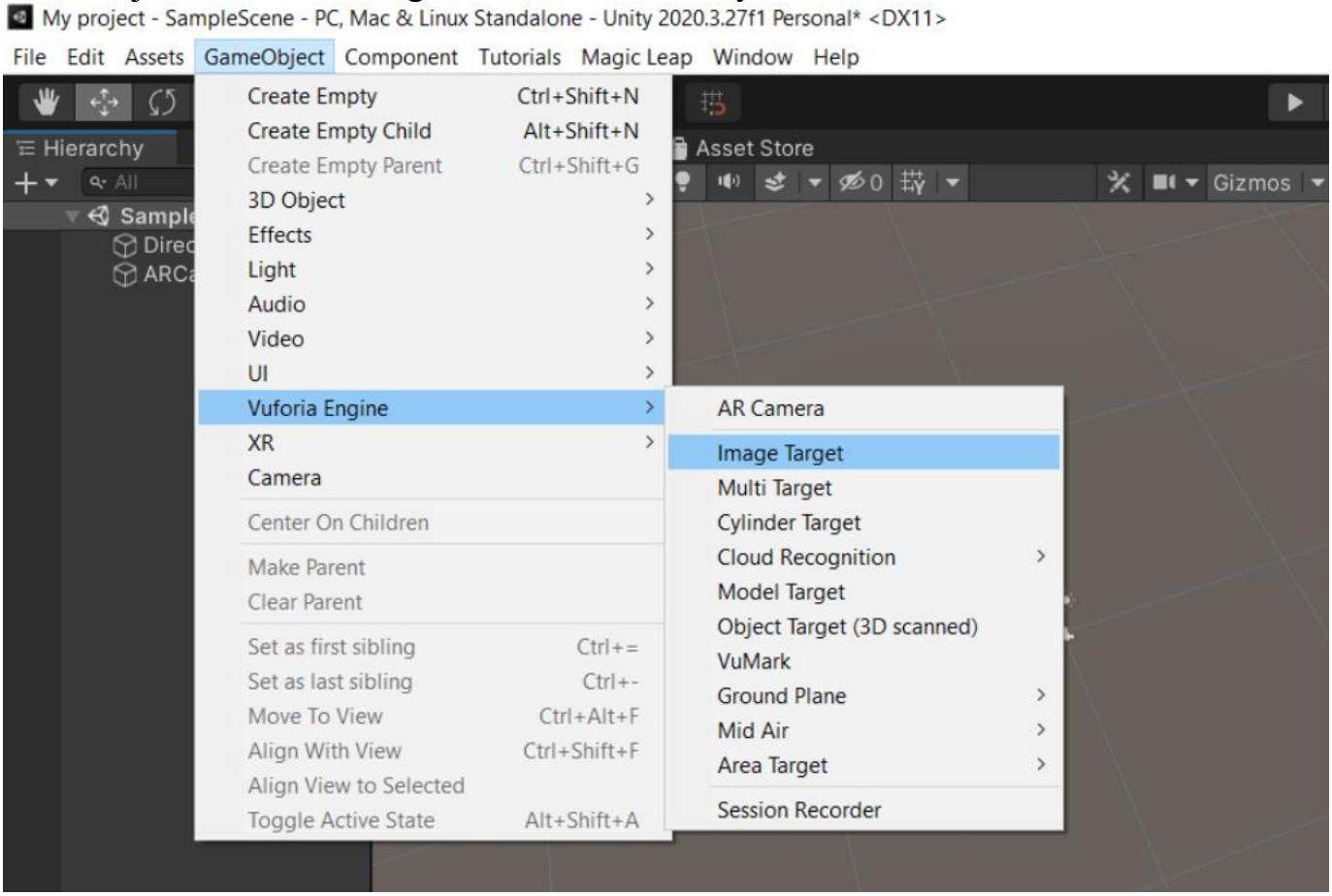


Рис.15. Вимкнення або видалення *CompleteOrderButton* в UI

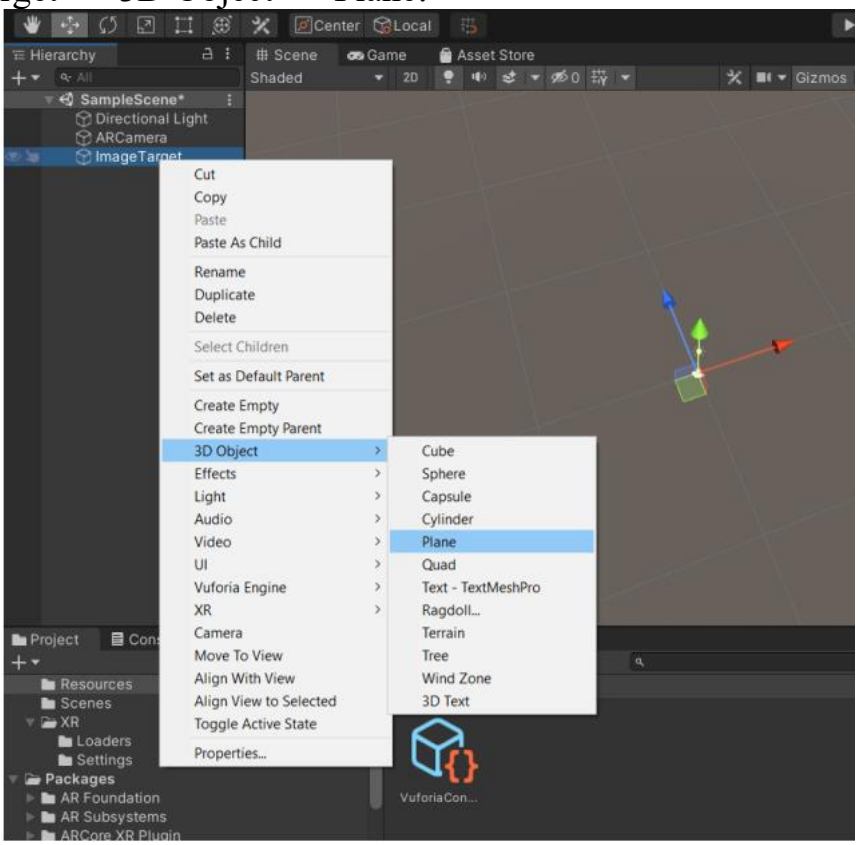
18. Збережіть сцену, натисніть *Play* та подайте піцу!
19. Скомпілювати два файли: під ОС Android 7.0 або вище та під ОС iOS 6.0 або вище.
20. Закомітити обидва файли та проект на гіт. Надати доступ своєму викладачу.

Завдання 2. Демонстрація відео в AR при наведенні камери на друге зображення-маркер

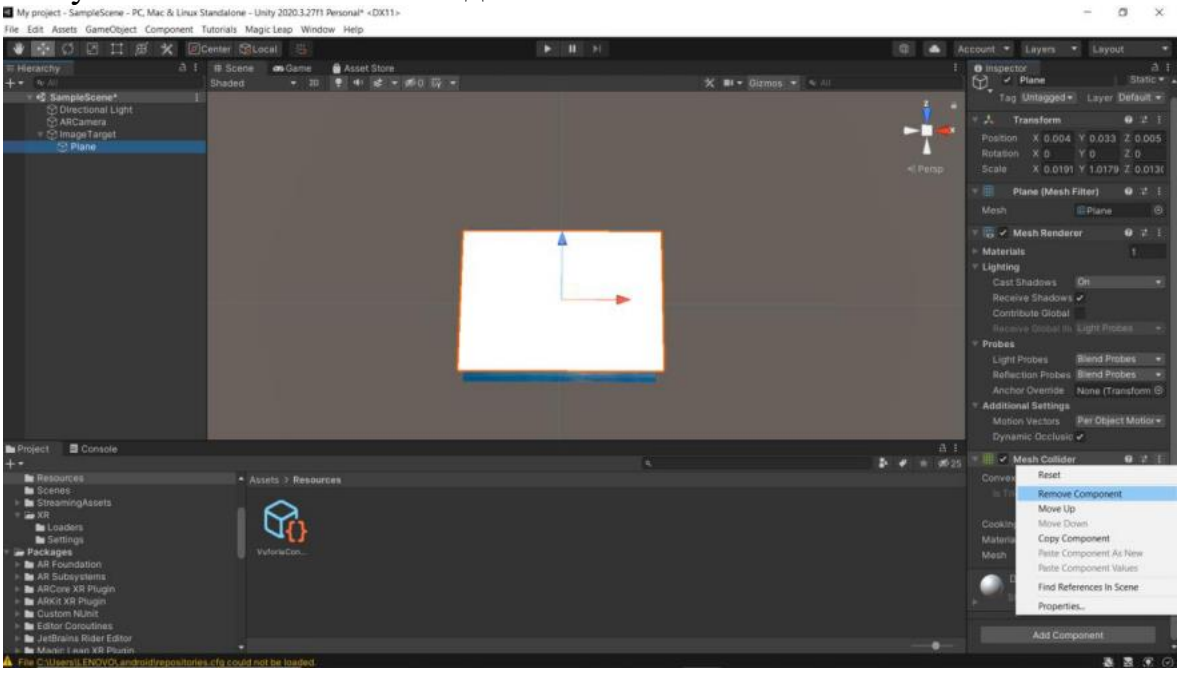
1. Додайте новий AR-проєкт, додайте AR-камеру та Image Target натиснувши GameObject >> Vuforia Engine >> «назва об’єкту».



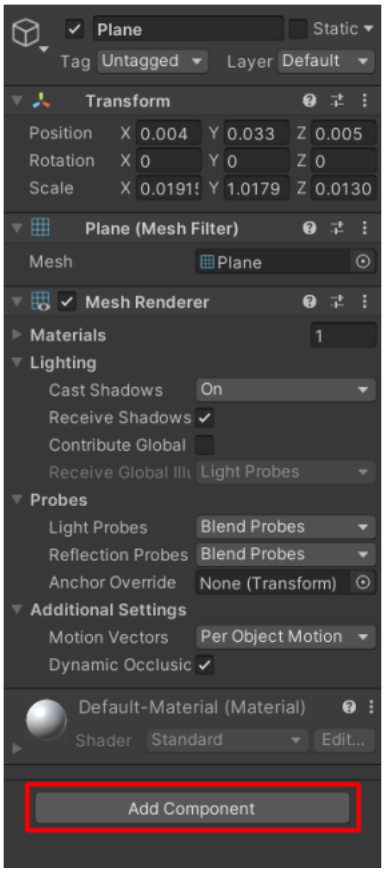
2. Додайте нову прощину або розмістіть на вже існуючій: правою клавішою миші на об’єкт ImageTarget >> 3D Object >> Plane.

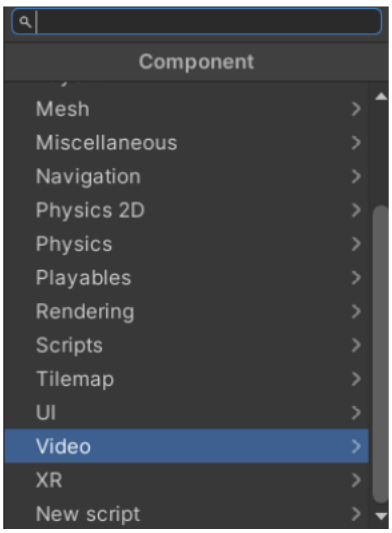


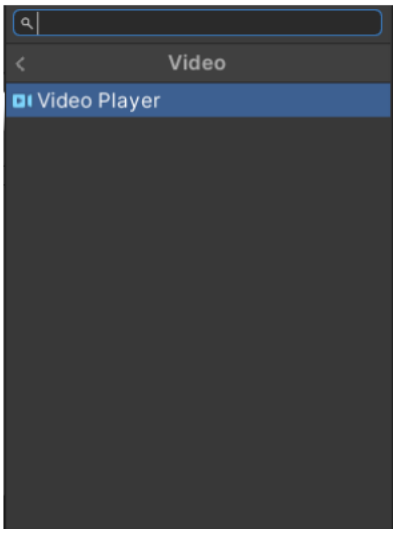
3. В налаштуваннях об’єкта Plane видаліть компонент Mash Collider.



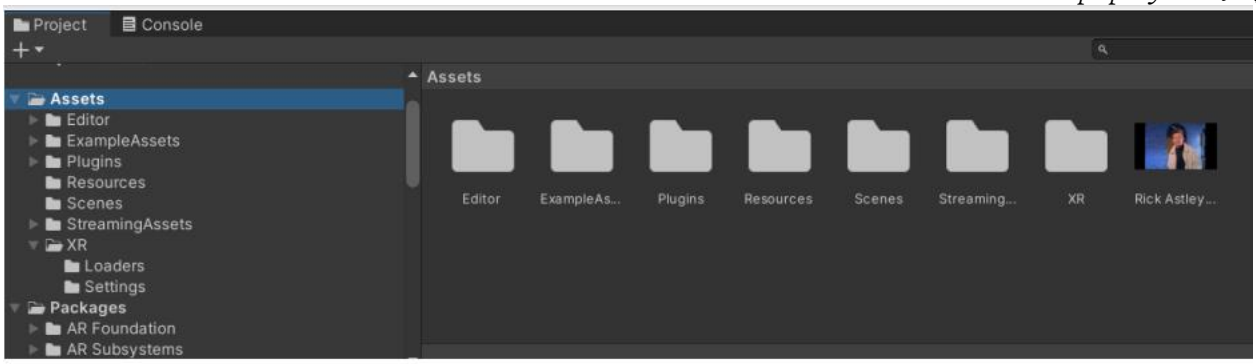
4. Додайте замість нього додаємо компонент Video Player.

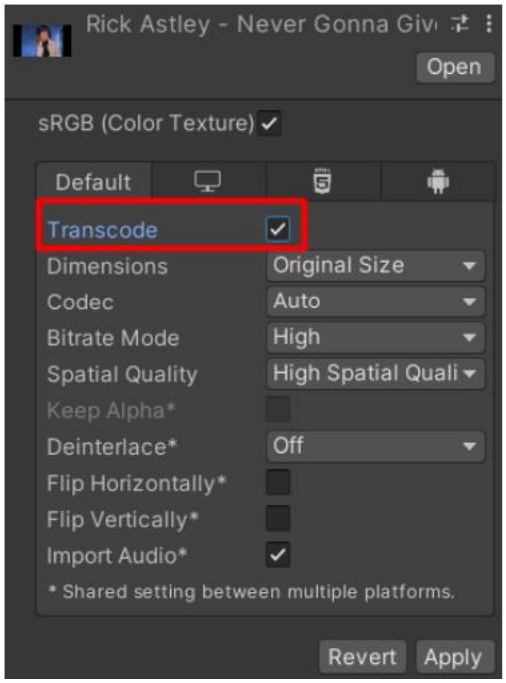




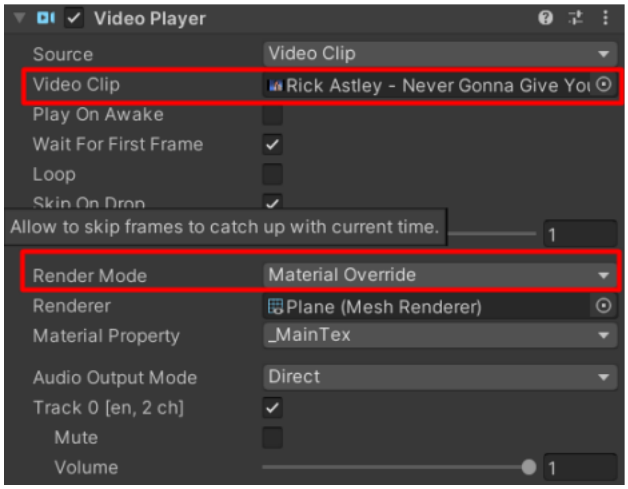


5. В папку Assets додайте відео та транскодьте його. Відео взяти довільно з інтернету з зображенням ректора.

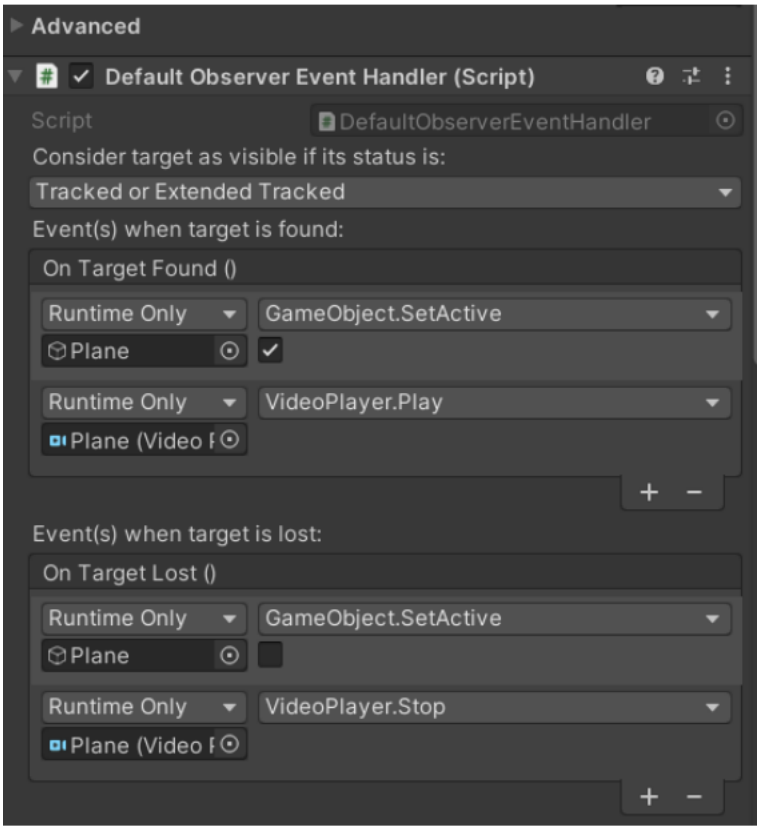




6. В налаштування компоненту Video Player об’єкта Plane додайту обране відео та оберіть Render Mode – Material Override.



7. В налаштуваннях об’єкта ImageTarget знайдіть компонент Default Observer Event Handler (Script) ті додайтк в поле On Target Found об’єкт Plane та методи GameObject.SetActive (зі значенням true – прапорець) та VideoPlayer.Play. В поле On Target Lost аналогічно додаємо методи GameObject.SetActive(false – немає галочки) та VideoPlayer.Stop.



8. Зібрати проєкт на Android, інсталювати його та перевірити працездатність. Закомітити проєкт та apk-файл на гіт, надати доступ Вашим викладачам.