DOI 10.34185/1562-9945-1-144-2023-08
UDC 004.946

V.V. Spirintsev, D.V. Sadychenko, O.V. Spirintseva

CROSS-PLATFORM UNITY APPLICATION FOR DISPLAYING 3D MODELS OF AUGMENTED REALITY USING ARCORE

Annotation. Recently, augmented reality technology has taken a qualitative step in development, which has enabled it to be useful in many areas of life. Augmented reality applications are unique in that they annotate or augment the user's reality. Practice has shown that augmented reality technology has unlimited potential and requires further research in the direction of creating innovative immersive applications. This work proposes a cross-platform Unity application for displaying 3D models of augmented reality using ARCore.
Keywords: augmented reality, ARCore, Unity, MVVM.

Formulation of the problem. Augmented reality is a term denoting all projects aimed at supplementing reality with any virtual elements in real time [1]. Recently, the technology of augmented reality has made a qualitative step in development, which has enabled it to be useful in many spheres of life (architecture, archeology, industry, artificial environment, educational environment, the field of entertainment, etc.) increasing the efficiency of the work carried out. ARCore technology originates from Tango, which is/was a more advanced AR toolkit that used special sensors built into the device. In ARCore, instead of special sensors, software (intended for Android devices) is used, which makes this technology more accessible and popular. ARCore can be accessed from multiple development platforms (Android [Java], Web [JavaScript], Unreal Engine [C++], and Unity [C#]), giving developers great flexibility and opportunities to create cross-platform applications. While each platform has its strengths and weaknesses, all platforms essentially extend from their own Android SDK. Augmented reality applications are unique in that they annotate or augment the user's reality. This is usually done visually, if an AR program superimposes computer graphics on the real world. Practice has shown [2] that augmented reality technology has unlimited potential and requires further research in the direction of creating innovative immersive applications.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Analysis of the latest research. Unity is a cross-platform game engine, which is mainly used for creating games, interactive 2D and 3D experiments (exercise simulation, medical and structural visualization), etc., developed by Unity Technologies [3]. Over the years, Unity has gradually introduced a set of tools to support 3D content creation, VFX and filmmaking. Embracing cross-platform development since its inception in 2006, Unity has allowed developers to compile their apps for multiple targets, making it a great tool for creating mobile games. Over the years, Unity has worked closely with Apple and Google to integrate the ARKit and ARCore SDK into its library. In 2018, Unity went public with a unifying API for both platforms - the AR Foundation package. AR Foundation allows you to create cross-platform augmented reality applications using Unity. Unity 2018.1 includes built-in cross-platform support for AR. These APIs reside in the UnityEngine.Experimental.XR namespace and consist of a number of subsystems. These subsystems contain low-level APIs for interacting with AR. The AR Foundation package wraps this low-level API into a coherent whole and enhances it with additional utilities such as managing the lifecycle of an AR session and creating GameObjects to represent discovered features in the environment. In the AR Foundation project, you choose which AR features to enable by adding the appropriate manager components to your scene. When you build and run your application on an AR device, the AR Foundation includes these features with its own AR SDK, so you can build it once and deploy it to the world's leading AR platforms [4].

The goal of the work. The purpose of this article is to develop a cross-platform Unity application for displaying 3D models of augmented reality using ARCore.

Main part. For the development of the "Room-Builder" application (Figure 1), the Unity platform and the C# programming language (for writing scripts) using the MVVM pattern (for building a high-quality connection between the interface and business logic, which in turn will help maintain a clean and clear application architecture). The following modules were additionally installed: Android SDK; Android NDK; ARCore foundation; ARCore XR.

To build the augmented reality of the ARCore technology, you need the following.

1. Track movement (Motion tracking) allows the smartphone to understand its position in the real world. This device needs data from accelerometer, magnetometer, compass and geolocation.

2. Environment understanding allows the smartphone to determine the size and position of all types of surfaces (vertical, horizontal and angular) and the distance from the device to these surfaces. Images from the device's cameras and laser distance sensors, if available (Figure 2), act as input data.

Figure 1 - Application interface

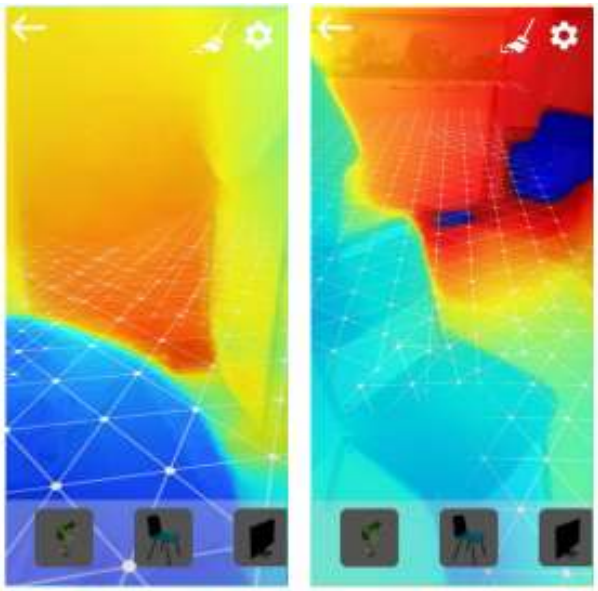

Figure 2 - Understanding image depth

3. Estimating the illumination allows the smartphone to estimate the illumination of the surrounding environment. Without this feature ARCore will not be able to build a plane with coordinates.

As an intermediate output we get a set of planes that contain coordinates and can be used to establish a model, track a model, or a point in space. We also get an opportunity to see the depth of the plane recognition, and see what ARCore sees and how it builds planes in the environment. The scheme of the organization of input and output data in the developed application is shown in Figure 3.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

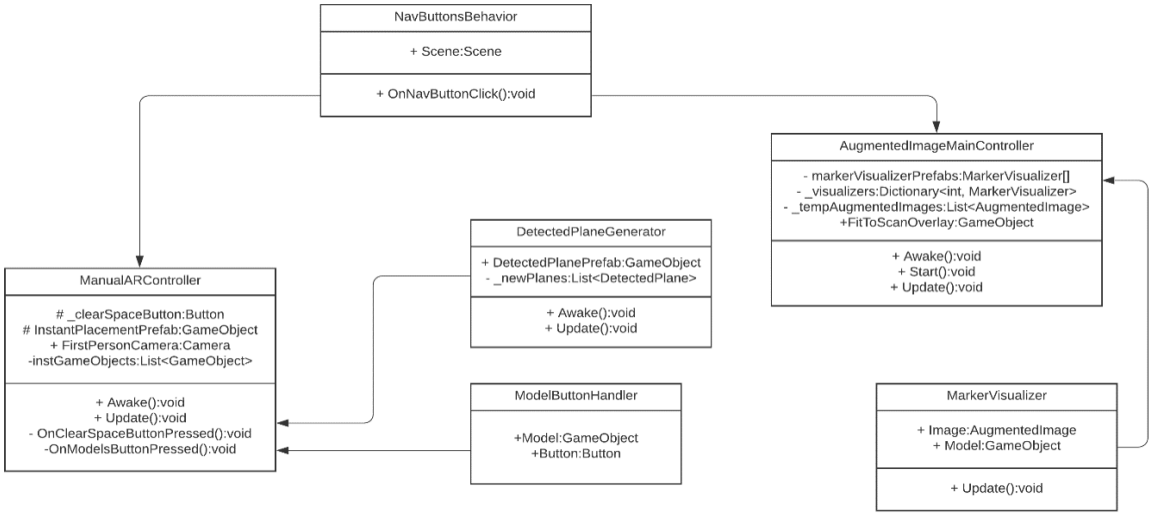

Figure 3 - Data processing scheme

Figure 4 shows the structure of the project.

Figure 4 - Structure of the developed application

The developed application has two work scenarios (see Figure 1). They are Manual placement of models and Tracking markers in the environment.

Manual placement of models. In the Unity editor, we will create a new scene for manual placement of models. Add a controller with an array of objects.

```
public class ModelButtonHandler : MonoBehaviour
{
    public GameObject Model;

    public Button Button;}
```

This model adds a construct to the Unity inspector to bind a button and a 3D object.

In the Update method, which is called with every update of the application frame, we will describe the logic of adding a model to the specified plane.

```
    if (hit.Trackable is DetectedPlane)
    {
    DetectedPlane detectedPlane = hit.Trackable as Detected-
Plane;
    if (detectedPlane.PlaneType == DetectedPlaneType.Vertical)
    prefab = InstantPlacementPrefab;
    }
    // Instantiate prefab at the hit pose.
    var gameObject = Instantiate(prefab, hit.Pose.position,
hit.Pose.rotation);
    instGameObjects.Add(gameObject);
```

It can be seen that under the condition of touching the specified ARCore plane, the selected 3D object is added to the surface and fixed at the initial coordinates. The Instantiate method allows you to create objects in a certain location and with a certain rotation. The diagram of the location of the models is shown in Figure 5.

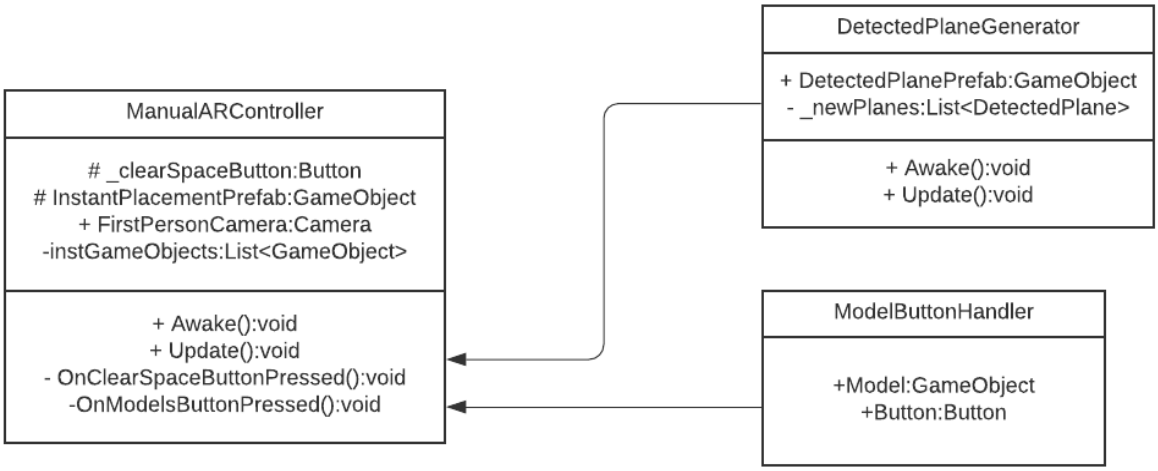

Figure 5 - Class diagram in the scene of manual arrangement of models

Figure 6 – Manual placement of models

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Tracking markers in the environment. In the Unity editor, we create a new scene for tracking markers and placing models on their surface. First, you need to add an empty Canvas and an ARCore Device object. At this stage, we add to the ARCore Device in the Markers Tracking Session inspector. It is a configuration file that is responsible for various parameters such as camera autofocus, face detection, image depth, and marker database. Next, we need to create a database with images for tracking. In the project window, select the desired set of control images (PNG or JPG), right-click and select Create → Google ARCore → AugmentedImageDatabase.

As in the previous example, we create a scene controller and add a C# script to it. In the Update() method, we define the software part of searching for images from the device's camera.

```
Session.GetTrackables<AugmentedImage>(_tempAugmentedImages, TrackableQueryFilter.Updated);
```

In the same script, we will define an array of 3D models, and attach them to images with a simple cyclic iterator.

```
public MarkerVisualizer[] markerVisualizerPrefabs;
```

If the recognition is successful, we add the model to the scene at zero coordinates (the center of the marker).

```
if (image.TrackingState == TrackingState.Tracking && marker == null && image.TrackingMethod == AugmentedImageTrackingMethod.FullTracking)

{

   Anchor anchor = image.CreateAnchor(image.CenterPose);
        marker = (MarkerVisualizer) Instantiate(markerVisualizerPrefabs[image.DatabaseIndex], anchor.transform);
        marker.Image = image;
        _visualizers.Add(image.DatabaseIndex, marker);
        FitToScanOverlay.SetActive(false);
}
```

Figure 7 shows the result of the marker tracking application.

Figure 7 - Label tracking

The developed application also has a mechanism for handling exceptional situations such as conditions of insufficient lighting; conditions of poor surface texture; conditions of ultra-fast movement of the camera of the device.

Conclusions. The work proposes a cross-platform Unity application for displaying 3D models of augmented reality using ARCore, which enables the user to understand how ARCore works (how ARCore sees the environment, how the process of recognizing and constructing the depth of the environment, placing or tracking geometric models using ARCore takes place). The developed application provides an intuitive, simple graphic interface for the most convenient user experience; the ability to choose geometric models for location; the ability to track markers for the location of geometric models; the ability to understand the environment. Further research should be conducted in the direction of expanding the description of the available elements responsible for configuring the application.

REFERENCES

1. Augmented reality// AR. [Electronic resource]. Access mode. –
URL: https://en.wikipedia.org/wiki/Augmented_reality.
2. Augmented Reality, AR//AR. [Electronic resource]. Access mode.–
URL:https://www.it.ua/knowledge-base/technology-innovation/dopolnennaja-realnost-ar.
3. Unity.[Electronic resource]. Access mode. – URL: https://unity.com/ru.
4. AR Foundation. [Electronic resource]. Access mode. –
URL: https://unity.com/ru/unity/features/arfoundation.

Received 16.01.2023.
Accepted 23.01.2023.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Кросплатформний Unity додаток для відображення 3D-моделей доповненої реальності засобами ARCore

За останній час технологія доповненої реальності зробила якісний крок у розвитку, що дало їй змогу бути корисною у багатьох сферах життя (архітектура, археологія, промисловість, штучне середовище, навчальне середовище, сфера розваг та ін.) підвищуючи ефективність здійснених робіт. Додатки доповненої реальності унікальні тим, що анотують або доповнюють реальність користувача, що підвищує коефіцієнт їх зацікавленості та надає відчуття безпосередньої взаємодії з навколишнім середовищем. Практика показала, що технологія доповненої реальності має необмежений потенціал та потребує подальших досліджень в напрямку створення інноваційних іммерсивних додатків. В даній роботі запропоновано кросплатформний Unity додаток для відображення 3D-моделей доповненої реальності засобами ARCore. Для розробки додатку "Room-Builder" була обрана платформа Unity та мова програмування C# (для написання скриптів) з використанням патерну MVVM (для побудови якісного зв'язку між інтерфейсом та бізнес логікою, що в свою чергу допоможе підтримувати чисту та зрозумілу архітектуру застосунку). Також додатково були встановлені наступні модулі: Android SDK; Android NDK; ARCore foundation; ARCore XR. Для управління роботою додатком і його вмістом було розроблено користувальницький інтерфейс, що надає можливості користувачу отримати повний контроль над системою. В додатку наявні два сценарії роботи (ручне розташування моделей; відстеження маркерів у навколишньому середовищі), а також механізм обробки виняткових ситуацій (умови недостатньої освітленості; умови поганої текстури поверхні; умови надшвидкого руху камери пристрою). Практична значимість даного програмного продукту полягає в тому, що створений додаток надає можливість користувачу зрозуміти як працює ARCore, побачити те - як ARCore бачить навколишнє середовище, як відбувається процес розпізнавання та побудови глибини середовища, розміщення або відстежування геометричних моделей засобами ARCore.

Спірінцев В'ячеслав Васильович - к.т.н., доцент, доцент кафедри програмного забезпечення комп'ютерних систем НТУ "Дніпровська політехніка".

Садиченко Данило Валерійович – магістр, НТУ "Дніпровська політехніка".

Спірінцева Ольга Володимирівна – к.т.н., доцент кафедри електронних обчислювальних машин Дніпровського національного університету імені Олеся Гончара.

Spirintsev Viacheslav Vasyliovych - candidate of technical sciences, associate professor, associate professor of computer system's software department of the Dnipro University of Technology.

Sadychenko Danylo Valeriyovych – master of the Dnipro University of Technology.

Spirintseva Olga Volodymyrivna - candidate of technical sciences, associate professor of Computer Systems Department of Oles Honchar Dnipro National University.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)