

---

# Lab 2 - Calibration

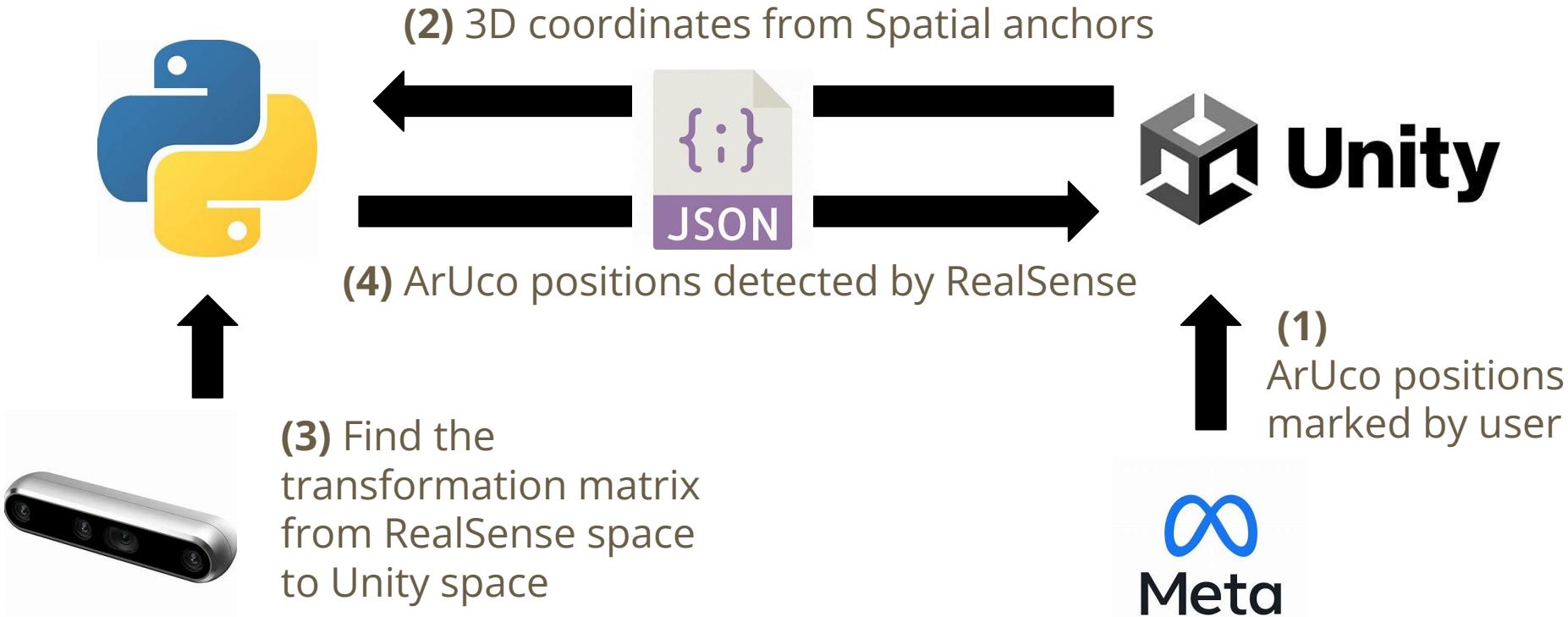
— Meta Quest 3 Spatial Anchor —

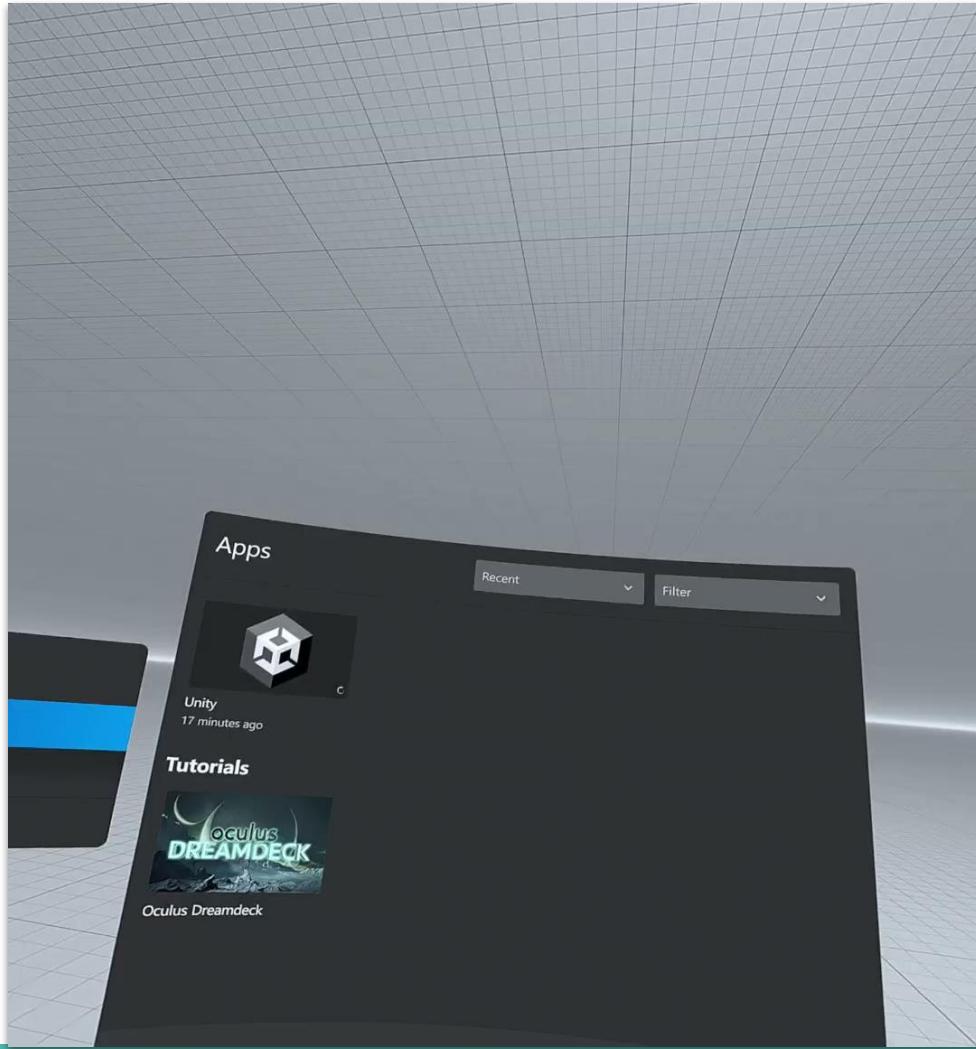
---

# Lab Schedule

- 9/12 Lab 1 - Marker Detection
- 9/19 Lab 2 - Calibration
- 9/26 Lab 3 - Game Design 1
- 10/3 Lab 4 - Game Design 2
- 10/10 National Holiday (No Class)
- 10/17 UIST (No Physical class)
- 10/24 Midterm Demo

# What we will do





# Outline

- Installation & Setup
- Camera & Controllers Tracking
- Spatial Anchor
- Calibration
- Server Client
- To Do

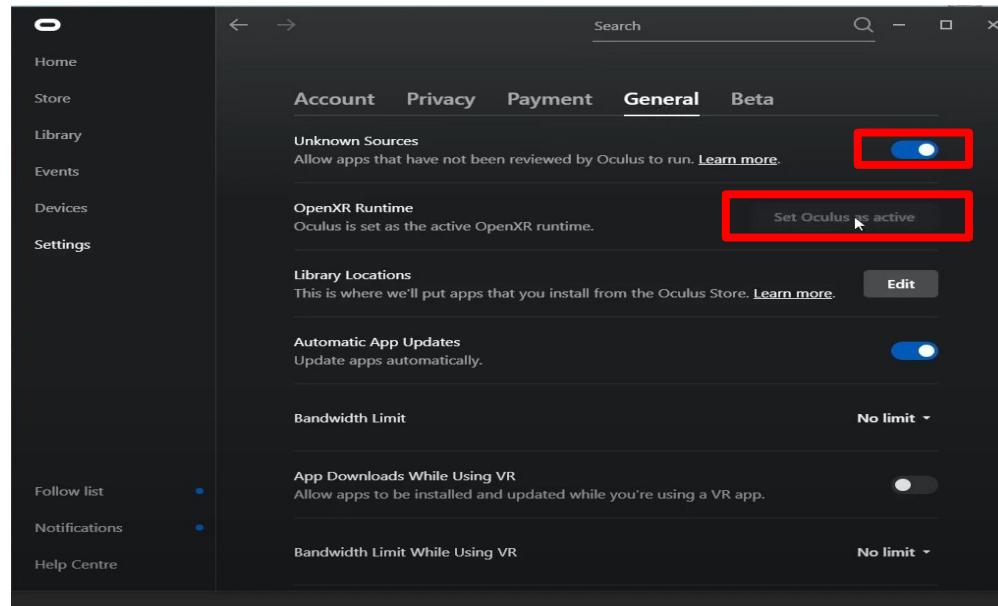
# Installation & Setup

# Quest Link

- Download Quest Link app and connect headset. [Link](#)

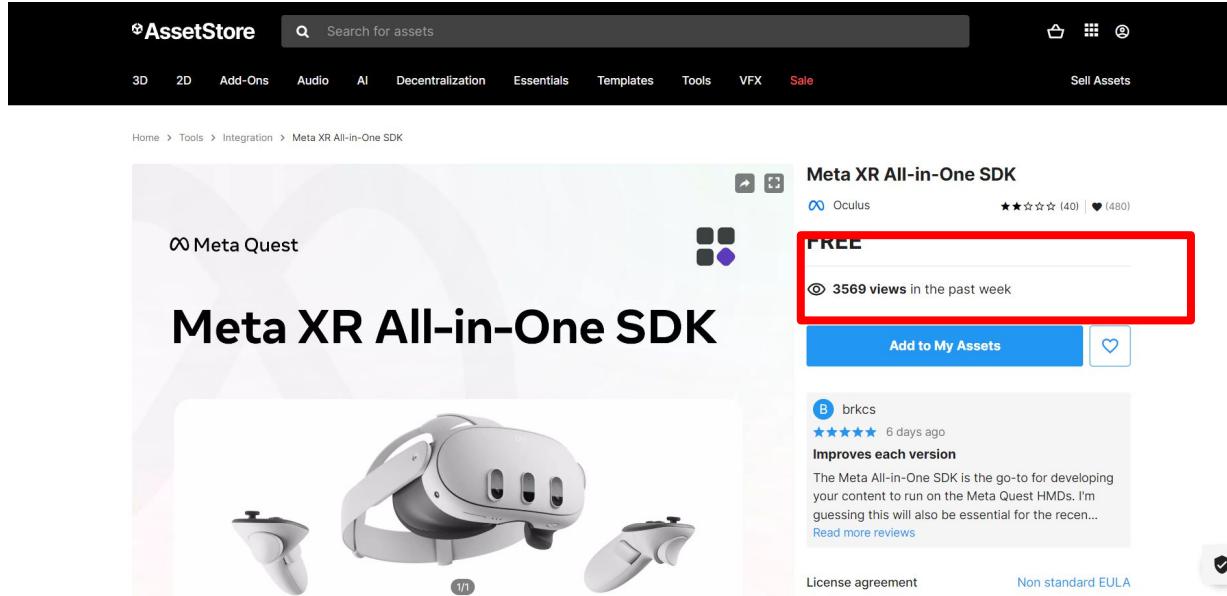
# Settings

- Go to Settings > General.
- Allow Unknown Source, and switch OpenXR Runtime to “Set Oculus as Active”.



# Meta XR All-in-One SDK [Link](#)

- Download and import to Unity project. [Link](#)



# Install Package

The screenshot shows the Unity Package Manager window. At the top left, there is a dropdown menu labeled "Packages: My Assets" with a red box around it. On the right side of the header, there is a large "Install" button for the selected package, also highlighted with a red box.

**Meta XR All-in-One SDK**  
60.0.0 · December 21, 2023 [Asset Store]

Oculus  
View in Asset Store | Publisher Website | Publisher Support

Description Version History Dependencies Images

Meta XR All-in-One SDK is a wrapper package that depends on the latest version of all main Meta XR SDKs, making it easy to get started with VR and MR development.

**Overview**

Purchased Date December 19, 2023

The Meta XR All-in-One SDK, com.meta.xr.sdk.all, bundles several Meta SDKs together, which includes many features that offer advanced rendering, social and community building, and provides capabilities to build immersive experiences in both virtual reality and mixed reality.

Looking for samples? All-in-One itself does not contain samples, but the dependencies it pulls in might - you can check samples on all the packages installed as a dependency in the Package Manager window by switching the "Packages" dropdown to "My Assets". Additionally, larger samples previously found in the Oculus Integrations package have been moved to github (<https://github.com/oculus-samples/Unity-StarterSamples>).

When installed, this package will pull in the following packages as dependencies:

- Meta XR Core SDK (<https://developer.oculus.com/downloads/package/meta-xr-core-sdk>)
- Meta XR Audio SDK (<https://developer.oculus.com/downloads/package/meta-xr-audio-sdk>)
- Meta XR Haptics SDK (<https://developer.oculus.com/documentation/unity/unity-haptics-sdk>)
- Meta XR Interaction SDK (<https://developer.oculus.com/downloads/package/meta-xr-interaction-sdk>)

# Fix and Apply

The screenshot shows the Oculus Project Setup Tool interface within a Unity Project Settings window. The left sidebar lists various project settings, with 'Oculus' selected. The main area displays a checklist of setup tasks.

**Oculus Project Setup Tool**  
This tool maintains a checklist of required setup tasks as well as best practices to ensure your project is ready to go. Follow our suggestions and fixes to quickly setup your project.  
Current project status: ! There are 3 outstanding Required fixes.

**Checklist**

Filter by Group : All

**Outstanding Issues (3)**

- ! The Oculus XR Plug-in package must be installed Fix
- ! The XR Plug-in Management package must be installed Fix
- ! Manual selection of Graphic API, favoring Direct3D11 Fix

**Recommended Items (5)**

- ! Color Space is required to be Linear Apply
- ! Disable Graphics Jobs Apply
- ! Use Stereo Rendering Instancing Apply
- ! Set maximum pixel lights count to 3 Apply
- ! Enable Anisotropic Filtering on a per-texture basis Apply

**Verified Items (33)**

Fix All Apply All

# Fix and Apply

Project Settings

Project Setup Tool

This tool maintains a checklist of required setup tasks as well as best practices ensure your project is ready to go. Follow our suggestions and fixes to quickly setup your project.

Current project status: ! There are 4 outstanding Required fixes.

Checklist

Filter by Group : All

Outstanding Issues (4)

- Either the Oculus XR (com.unity.xr.oculus) or OpenXR Plugin (com.unity.xr.openxr) package must be installed through the Unity Package Manager.
- The XR Plug-in Management (com.unity.xr.management) package must be installed through the Unity Package Manager.
- Minimum Android API Level must be at least 29
- Use ARM64 as target architecture

Fix All

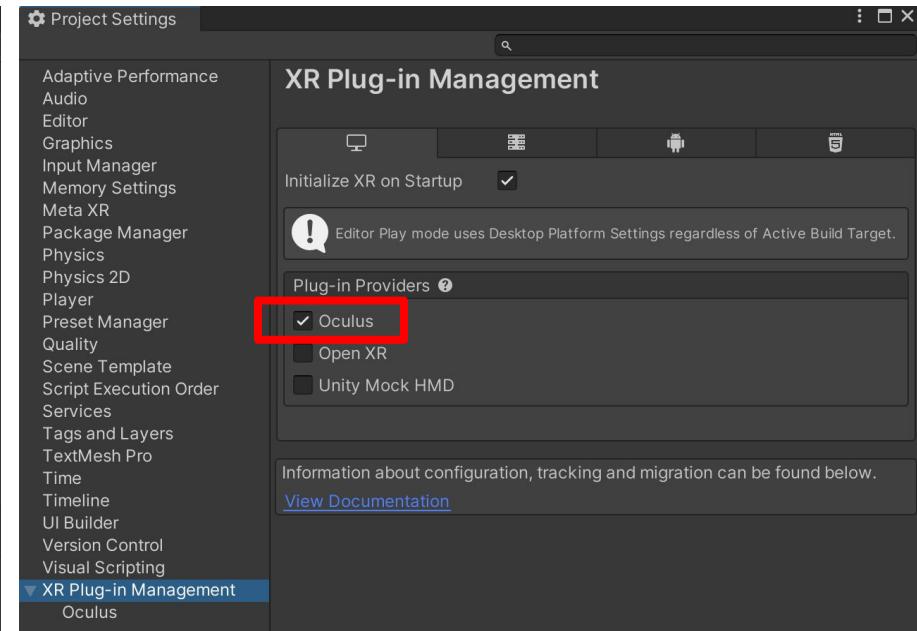
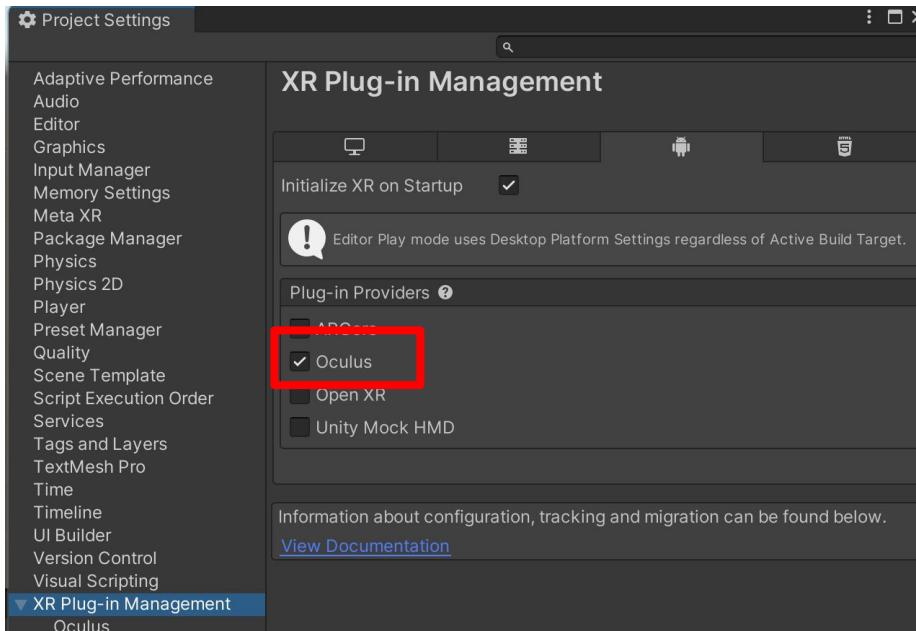
Recommended Items (6)

- Using IL2CPP as the scripting backend is recommended.

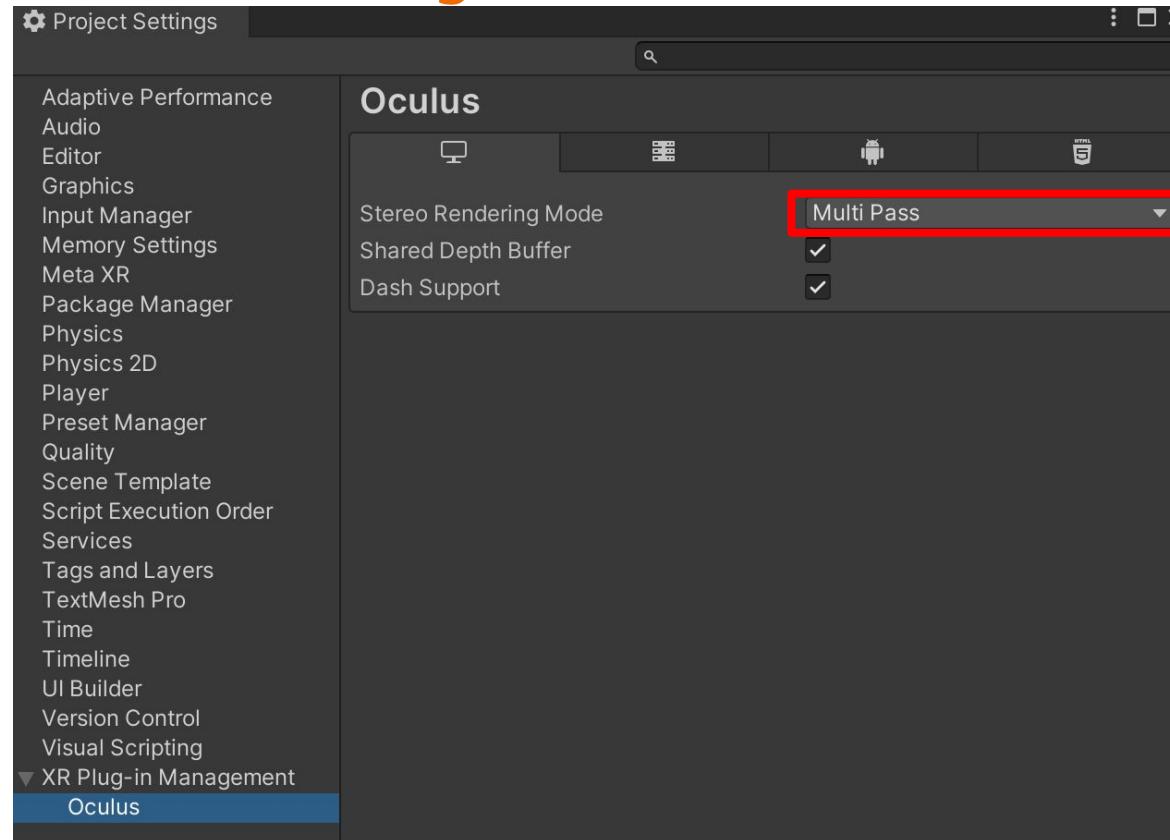
Apply All

The screenshot shows the Project Setup Tool window in Unity. On the left is a sidebar with various settings categories. The 'Meta XR' category is selected and highlighted in blue. The main area displays a checklist titled 'Outstanding Issues (4)' with four items listed. Below this is a section titled 'Recommended Items (6)' with one item listed. Two buttons are prominently displayed: 'Fix All' (overlaid on the 'Outstanding Issues' section) and 'Apply All' (overlaid on the 'Recommended Items' section). Both of these buttons are enclosed in red rectangles, indicating they are the focus of the 'Fix and Apply' process.

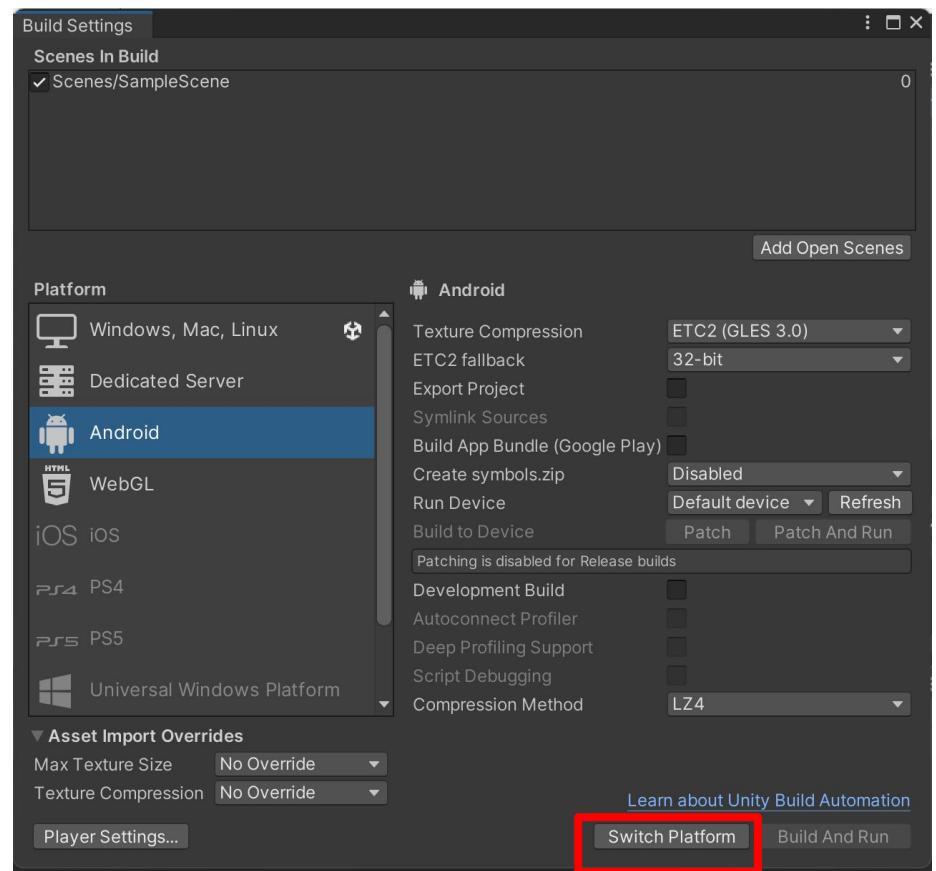
# XR Plug-in Management



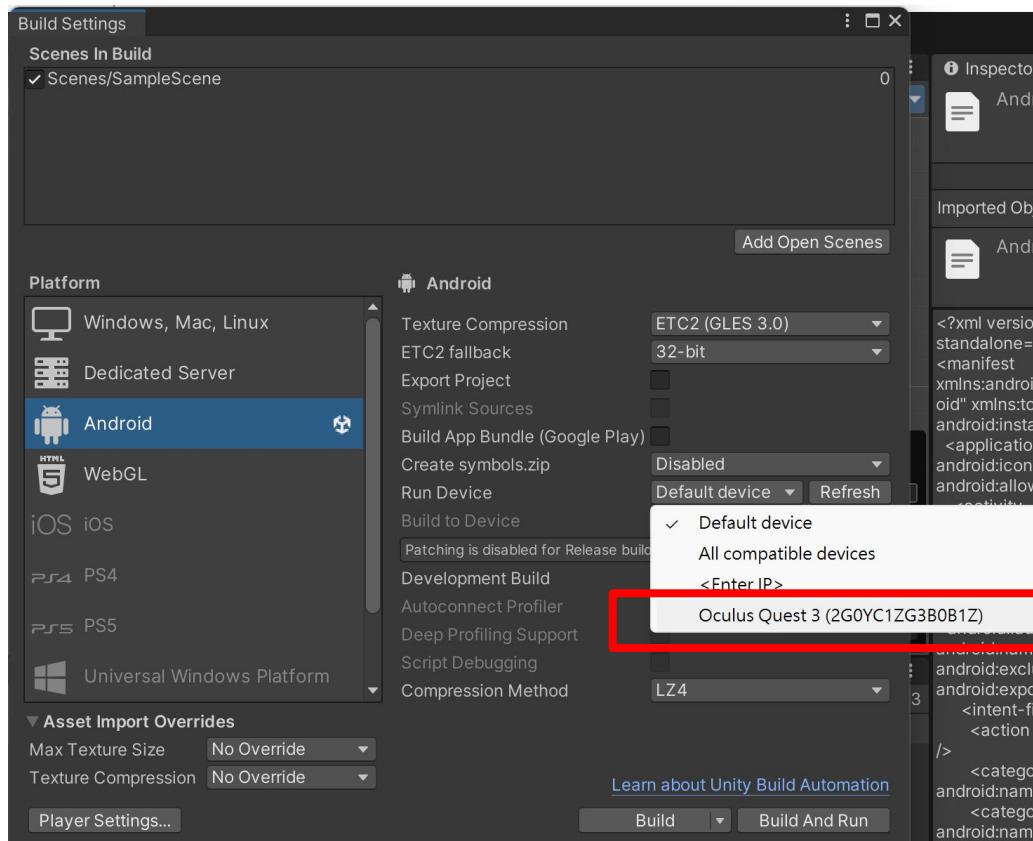
# Change Stereo Rendering Mode



# Switch to Android Platform

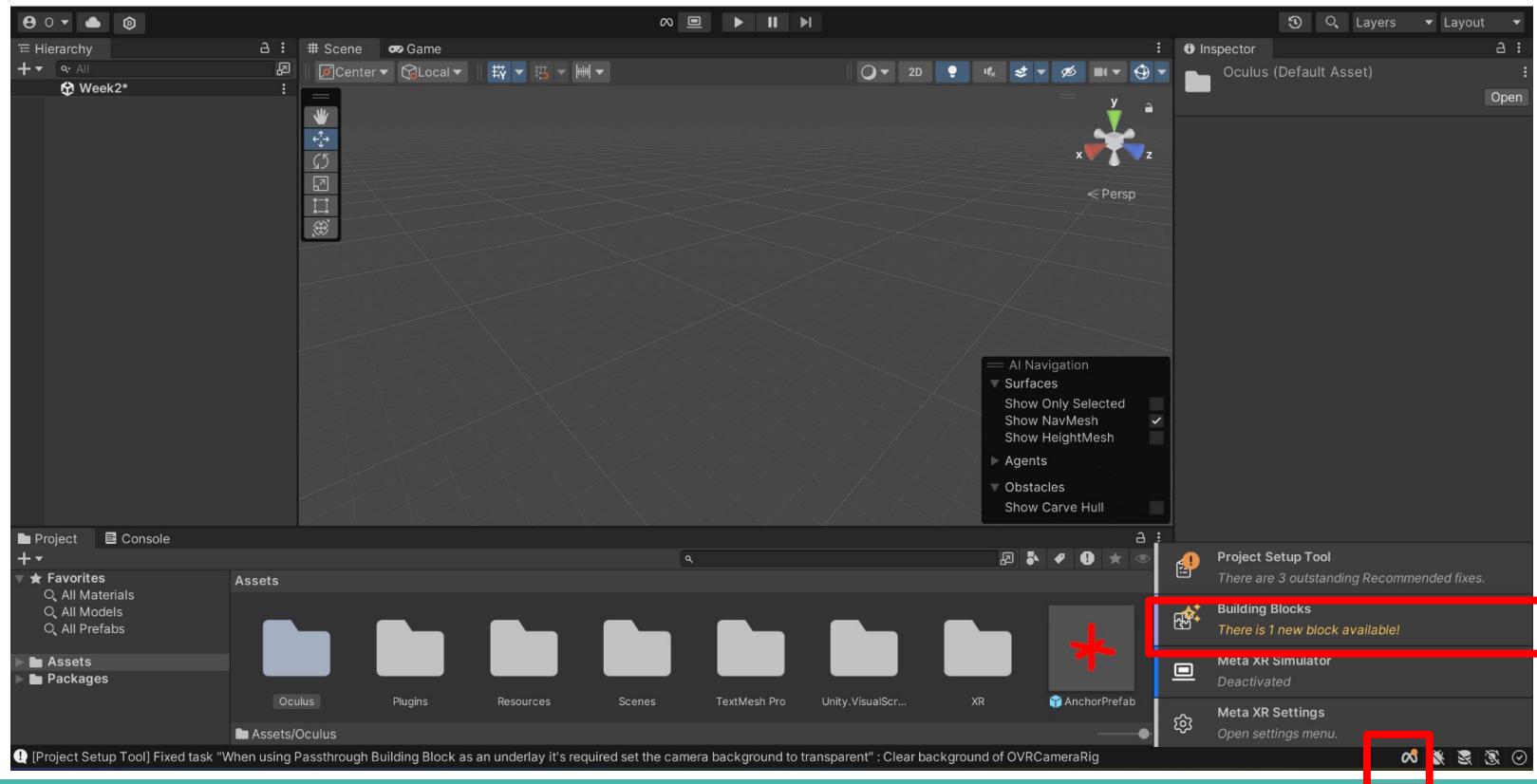


# Refresh “Run Device”and select Quest 3

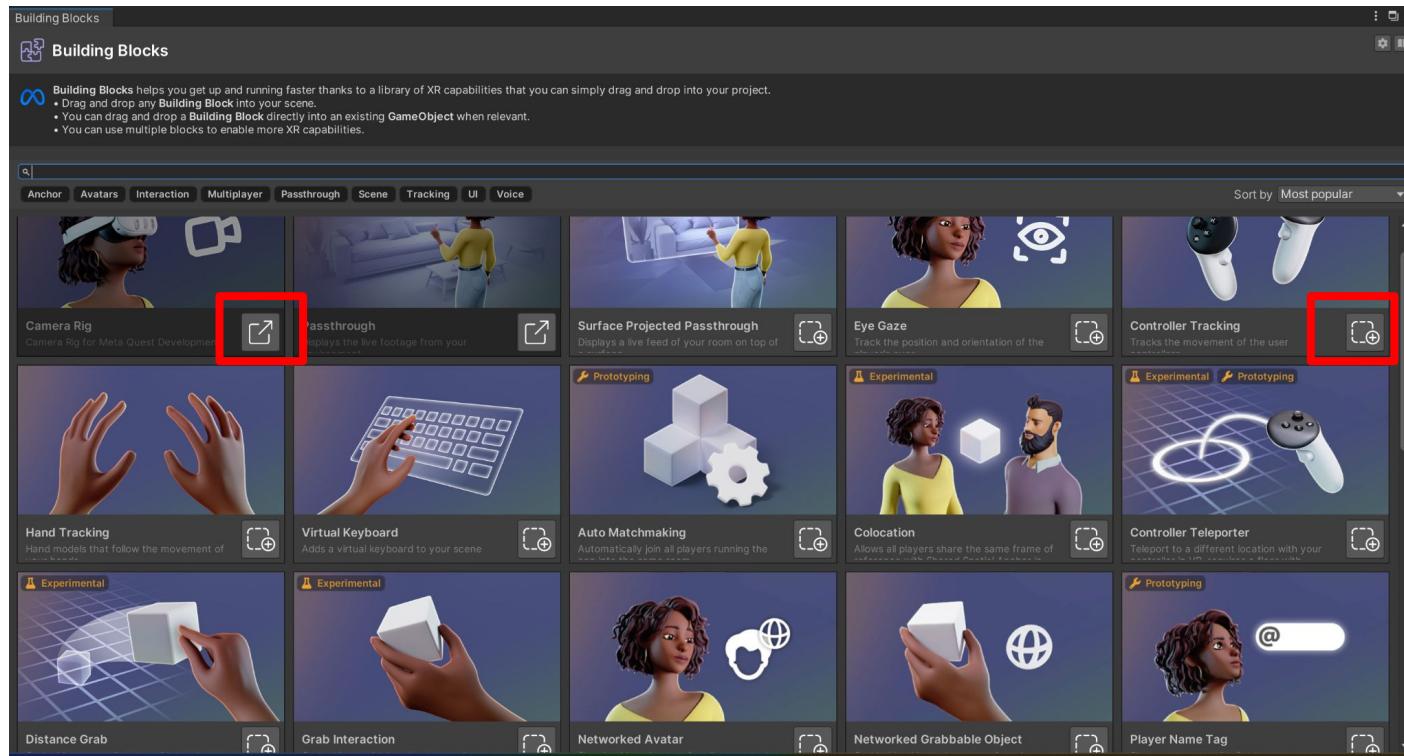


# Camera & Controllers Tracking

# Meta Building Blocks



# Select Camera Rig and Controller Tracking

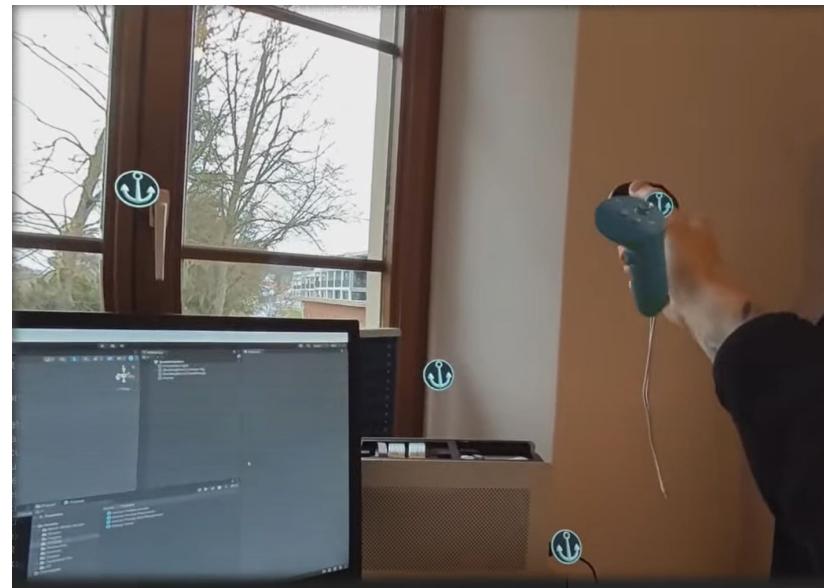




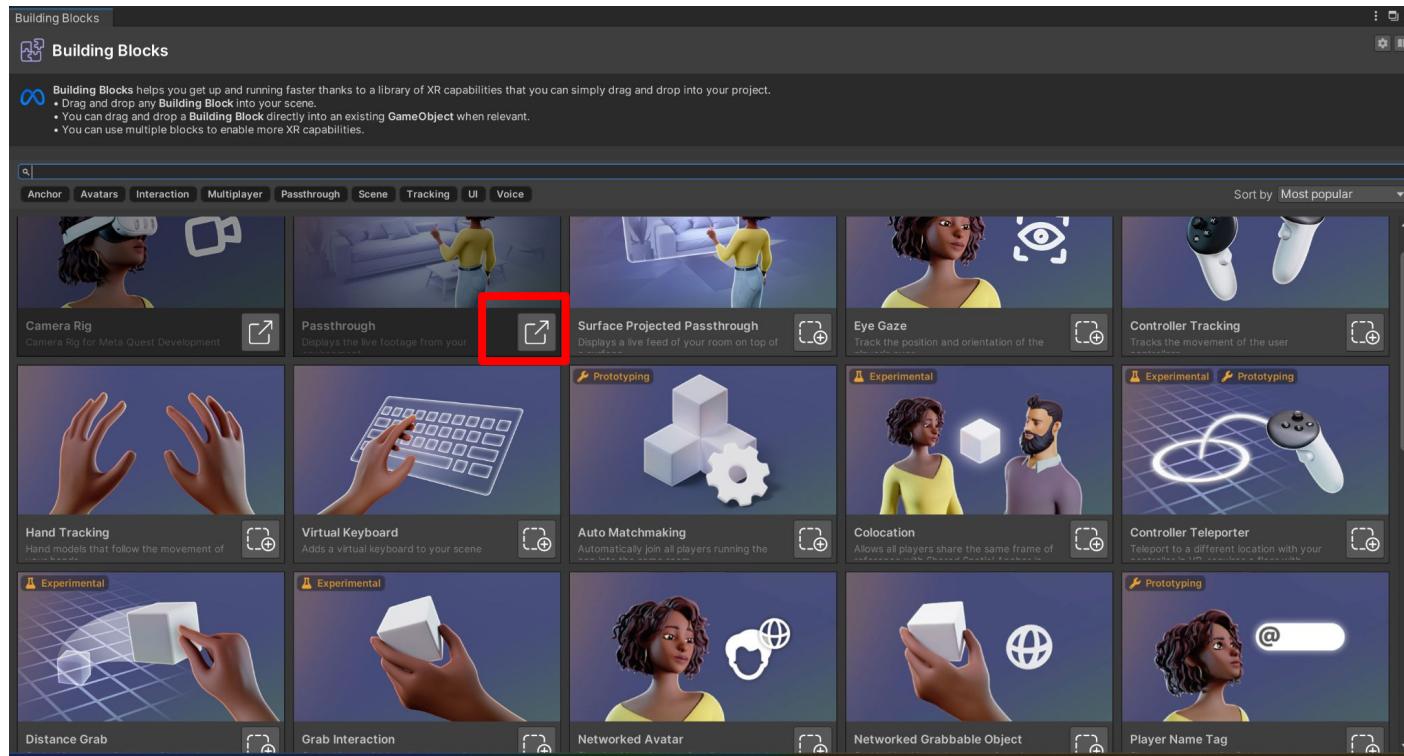
# Spatial Anchor

# Spatial Anchors

- Stable points in real world tracked by Quest 3



# Add Passthrough



# SpatialAnchors.cs

```
7  public class SpatialAnchors : MonoBehaviour
8  {
9      //Specify controller to create Spatial Anchors
10     [SerializeField] private Controller controller;
11     // Spatial Anchor Prefab
12     public GameObject anchorPrefab;
13
14     // Update is called once per frame
15     void Update()
16     {
17         // Create Anchor when user press the index trigger on specified controller
18         if(OVRInput.GetDown(OVRInput.Button.PrimaryIndexTrigger, controller))
19         {
20             CreateSpatialAnchor();
21         }
22     }
23
24     public void CreateSpatialAnchor()
25     {
26         GameObject anchor = Instantiate(anchorPrefab, OVRInput.GetLocalControllerPosition(controller),
27                                         OVRInput.GetLocalControllerRotation(controller));
28         anchor.AddComponent<OVRSpatialAnchor>();
29     }
30 }
```

| 搜尋文件                     | Q        | 已選擇 0 個項目 | + 資料夾     | ↑ 上傳 |
|--------------------------|----------|-----------|-----------|------|
| 名稱                       | 建立日期     | 修改日期      | 修改者       | 大小   |
| AnchorPrefab.prefab      | am 11:22 | am 11:22  | 22 KB     | ✓    |
| Client.py                | am 11:22 | am 11:22  | 768 bytes | ✓    |
| MoveByController.cs      | am 11:22 | am 11:22  | 738 bytes | ✓    |
| Server.cs                | am 11:22 | am 11:22  | 3 KB      | ✓    |
| <b>SpatialAnchors.cs</b> | am 11:22 | am 11:22  | 2 KB      | ✓    |

# OVRInput

- OVRInput.Get()
  - Return true if currently pressed.
- OVRInput.GetDown()
  - Return true if pressed this frame.
- OVRInput.GetUp()
  - Return true if released this frame.

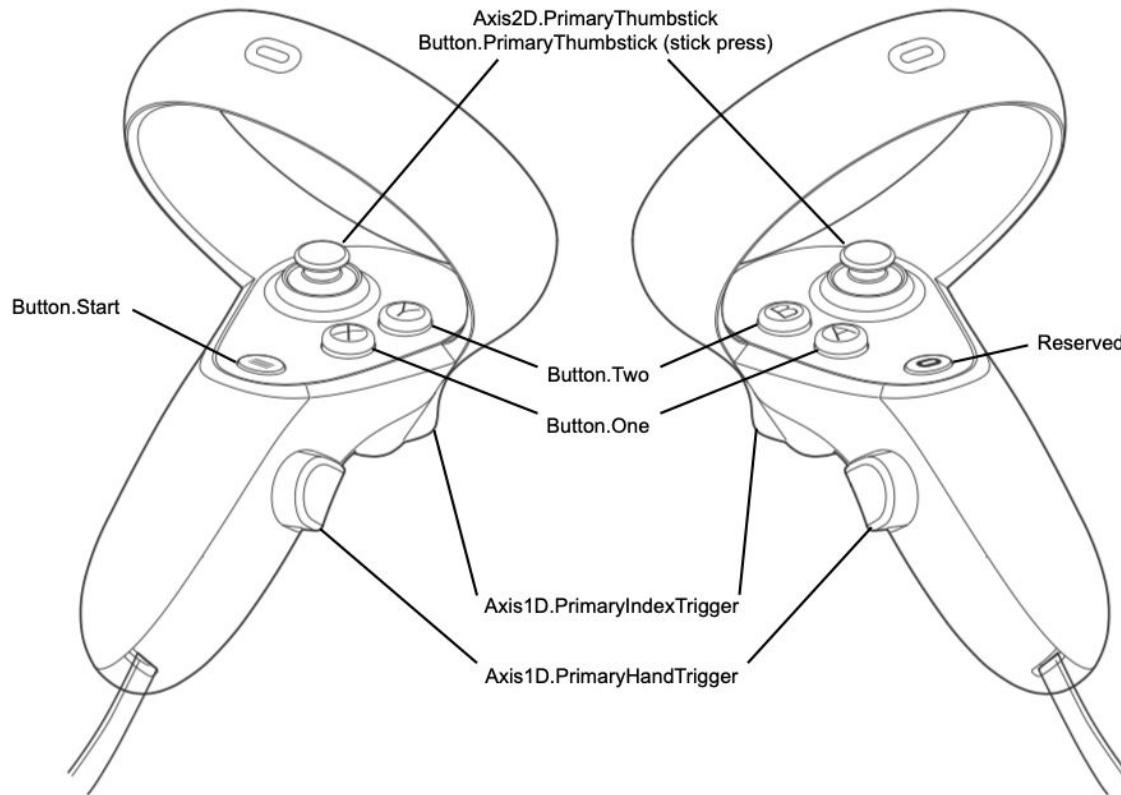
```
14 // Update is called once per frame
0 references
15 void Update()
16 {
17     // Create Anchor when user press the index trigger on specified controller
18     if OVRInput.GetDown(OVRInput.Button.PrimaryIndexTrigger, controller)
19     {
20         CreateSpatialAnchor();
21     }
22 }
23 }
```

# Control Input Enumerations

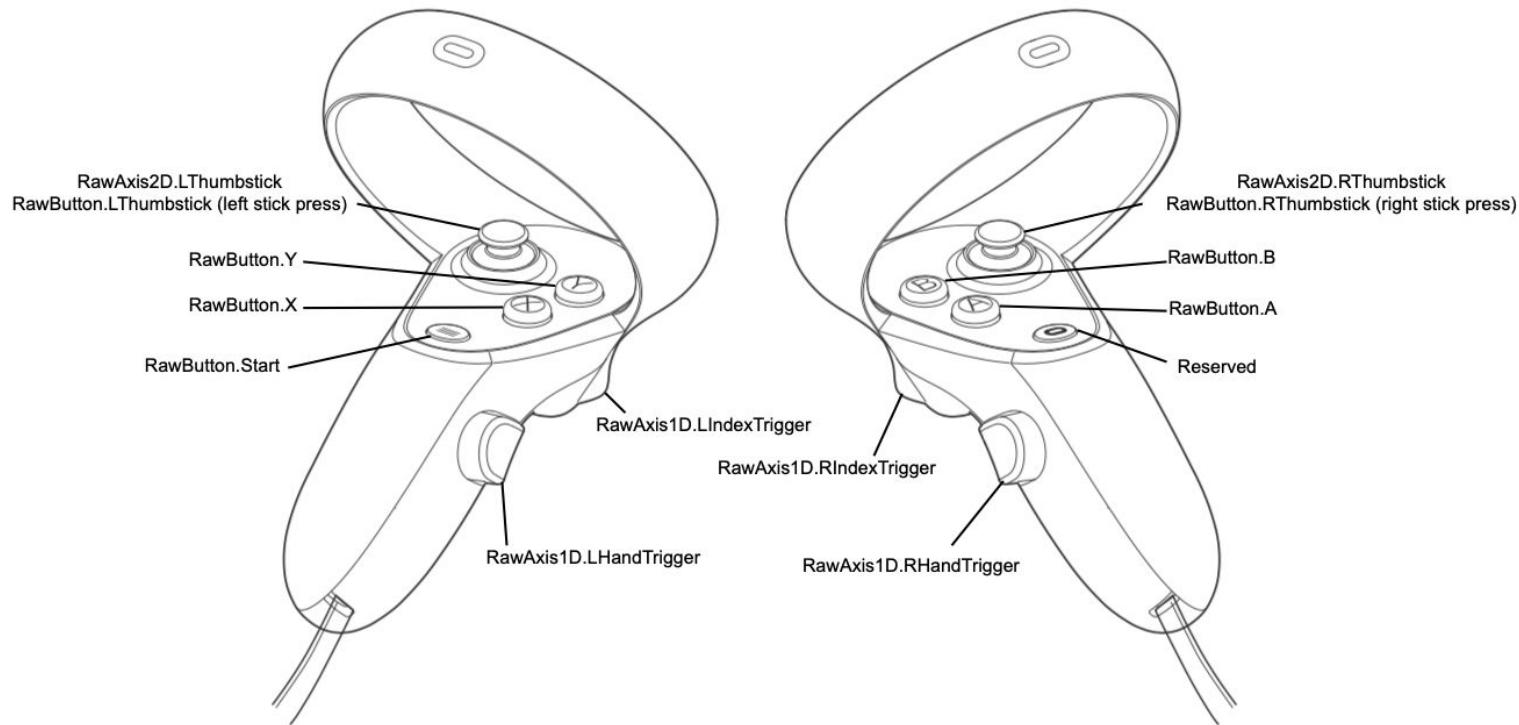
---

| Control            | Enumerates  |
|--------------------|---|
| OVRInput.Button    | Traditional buttons found on gamepads, controllers, and back button.          |
| OVRInput.Touch     | Capacitive-sensitive control surfaces found on the controller.                |
| OVRInput.NearTouch | Proximity-sensitive control surfaces found on the controller.                 |
| OVRInput.Axis1D    | One-dimensional controls such as triggers that report a floating point state. |
| OVRInput.Axis2D    | Two-dimensional controls including thumbsticks.<br>Reports a Vector2 state.   |

# Controller Input Mapping



# Controller Input Mapping (Raw)



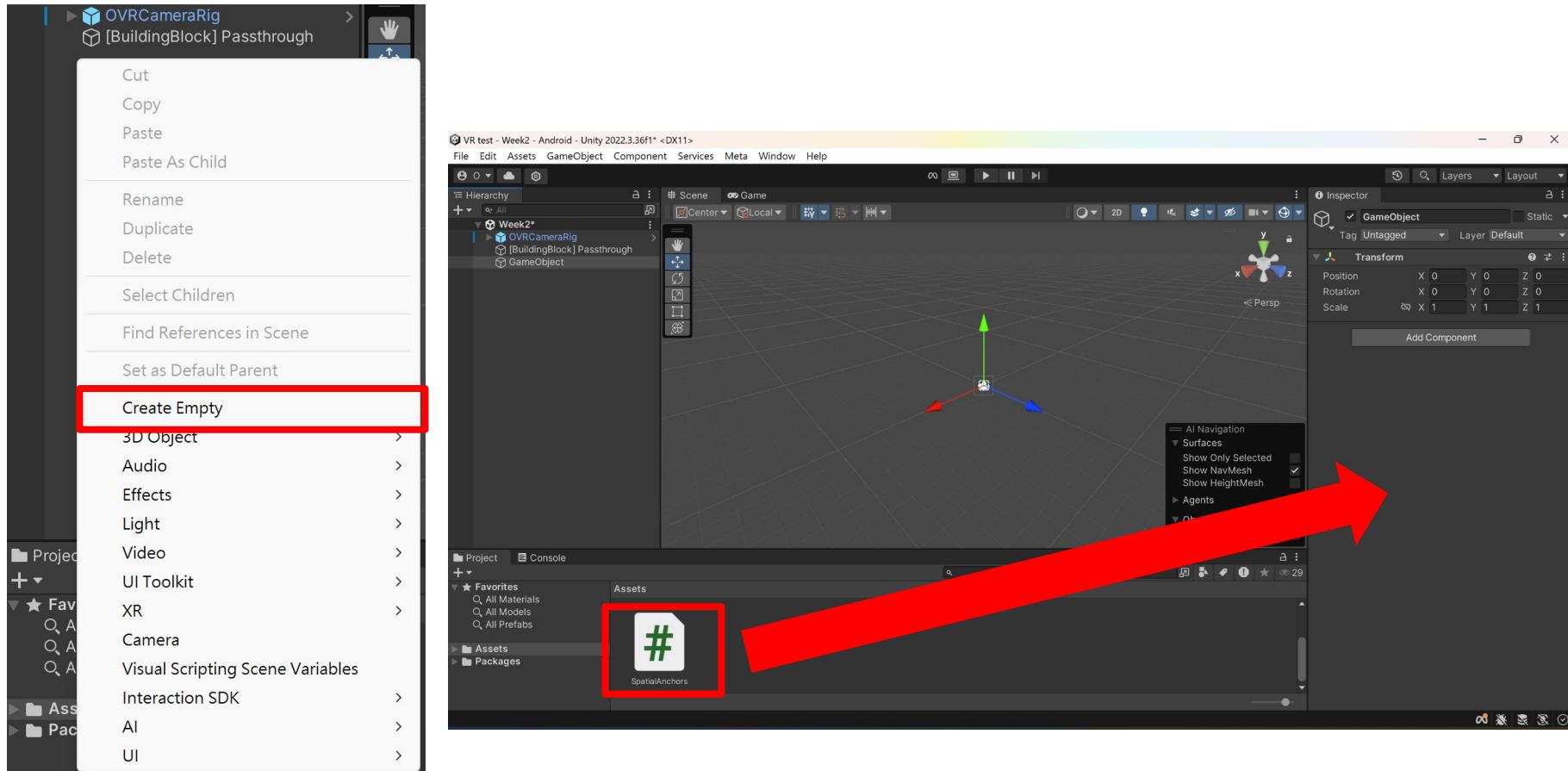
# Instantiate

Instantiate(Object original, Vector3 position, Quaternion rotation, Transform parent)

Instantiate(Object original, Vector3 position, Quaternion rotation)

```
24     public void CreateSpatialAnchor()
25     {
26         GameObject anchor = Instantiate(anchorPrefab, OVRInput.GetLocalControllerPosition(controller)
27                                         , OVRInput.GetLocalControllerRotation(controller));
28         anchor.AddComponent<OVRSpatialAnchor>();
29     }
```

|                                |  |
|--------------------------------|--|
| <b>original</b>                | An existing object that you want to make a copy of.  |
| <b>position</b>                | Position for the new object.   |
| <b>rotation</b>                | Orientation of the new object.   |
| <b>parent</b>                  | Parent that will be assigned to the new object.  |
| <b>instantiateInWorldSpace</b> | When you assign a parent Object, pass true to position the new object directly in world space. Pass false to set the Object's position relative to its new parent. |



搜尋文件  已選擇 0 個項目  + 資料夾  並上傳

• 互動系統設計與實作 Interactive S

• Lab 0

• Lab 1

• Lab 2

• Lab 3

• Lab 4

AnchorPrefab.prefab am 11:22 am 11:22 22 KB

Client.py am 11:22 am 11:22 768 bytes

MoveByController.cs am 11:22 am 11:22 738 bytes

Server.cs am 11:22 am 11:22 3 KB

SpatialAnchors.cs am 11:22 am 11:22 2 KB



AnchorPrefab

**Inspector**

**SpatialAnchor**  Static

Tag Untagged Layer Default

**Transform**

Position X 0 Y 0 Z 0

Rotation X 0 Y 0 Z 0

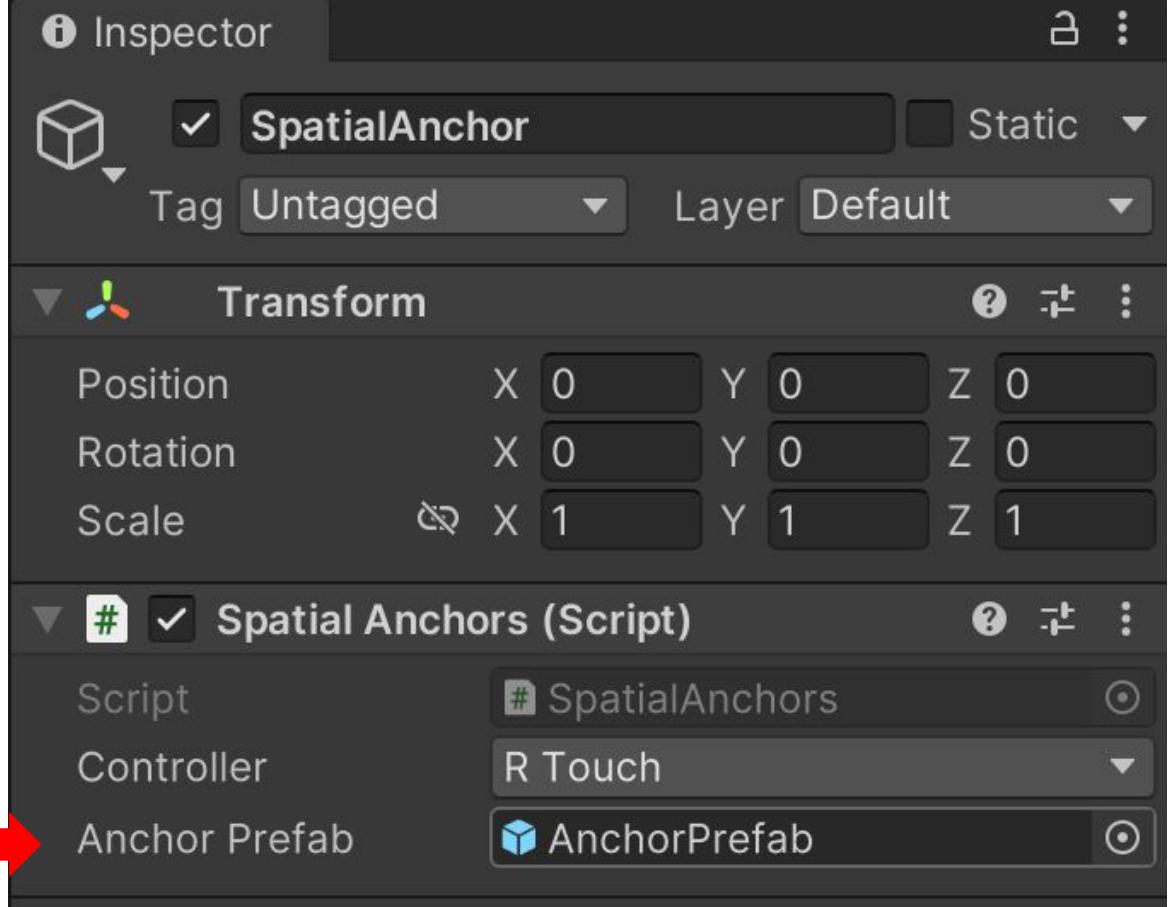
Scale X 1 Y 1 Z 1

**Spatial Anchors (Script)**

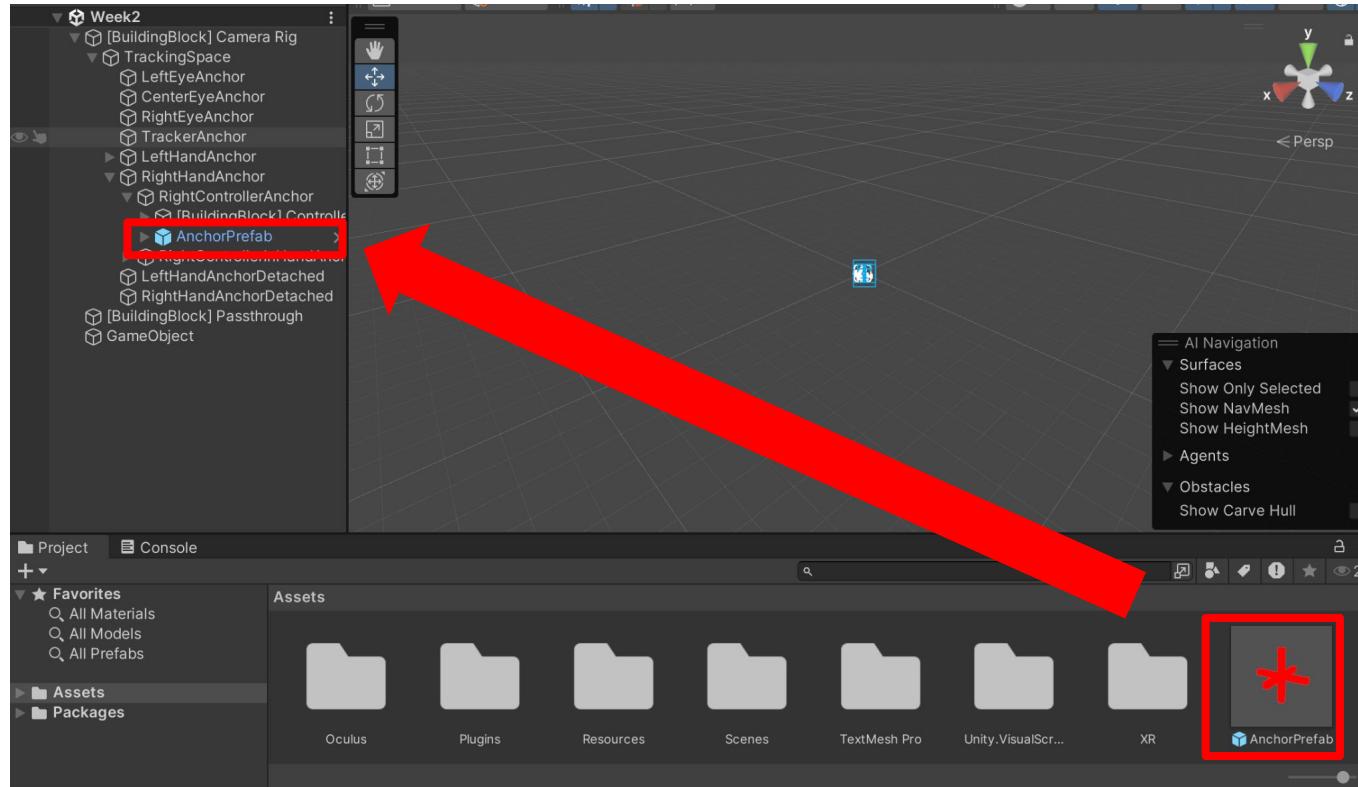
Script # SpatialAnchors

Controller R Touch

Anchor Prefab AnchorPrefab

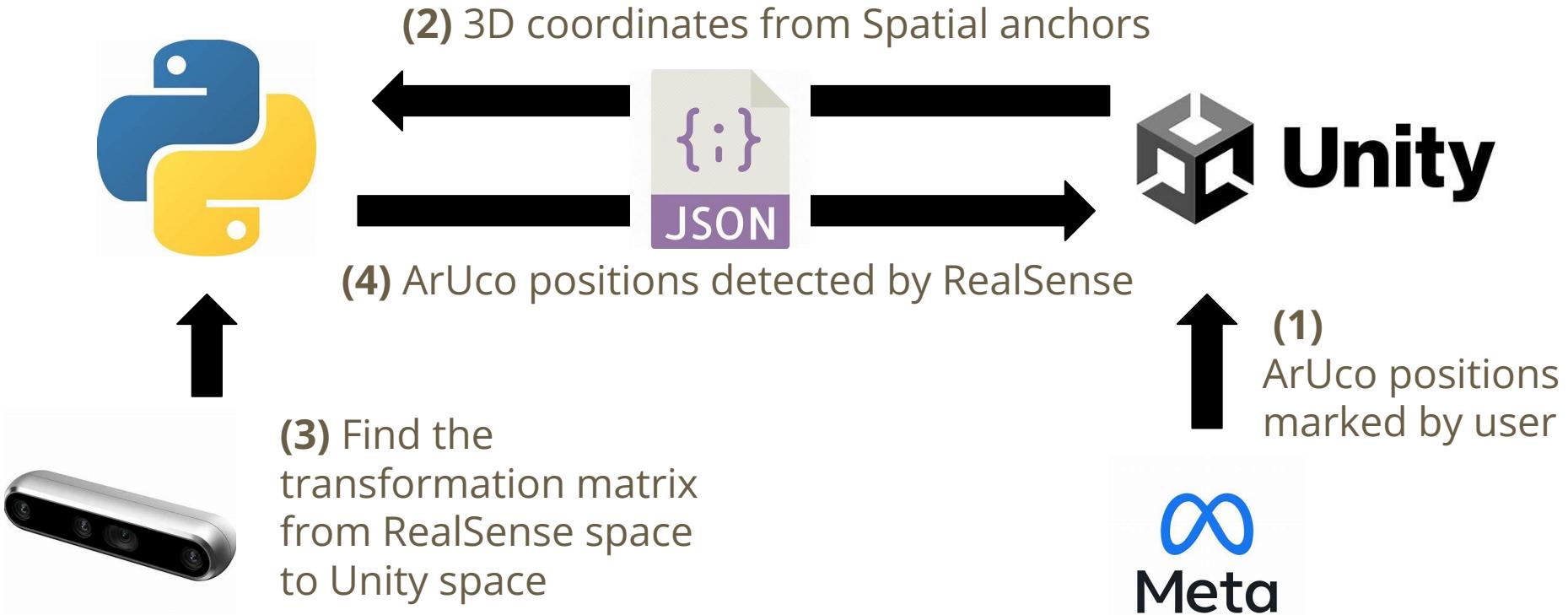


# Indicate where the anchor will be create.



# Calibration

# Calibration Architecture



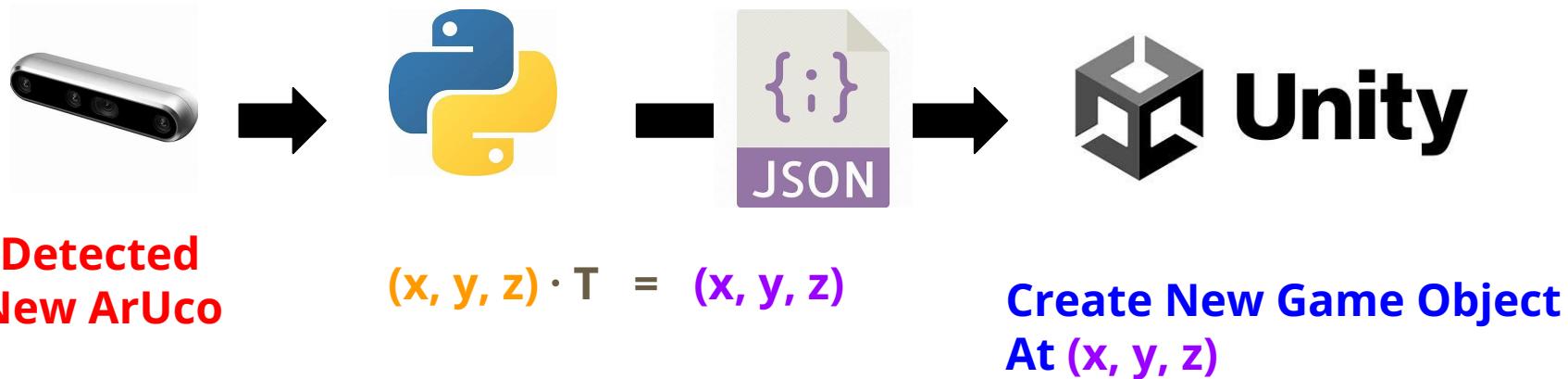
# Finding transformation matrix



|         | RealSense   | Unity                 | Meta             |
|---------|-------------|-----------------------|------------------|
| ArUco 1 | $(x, y, z)$ | $\cdot T = (x, y, z)$ | Spatial Anchor 1 |
| ArUco 2 | $(x, y, z)$ | $\cdot T = (x, y, z)$ | Spatial Anchor 2 |
| ArUco 3 | $(x, y, z)$ | $\cdot T = (x, y, z)$ | Spatial Anchor 3 |
| ArUco n | $(x, y, z)$ | $\cdot T = (x, y, z)$ | Spatial Anchor n |

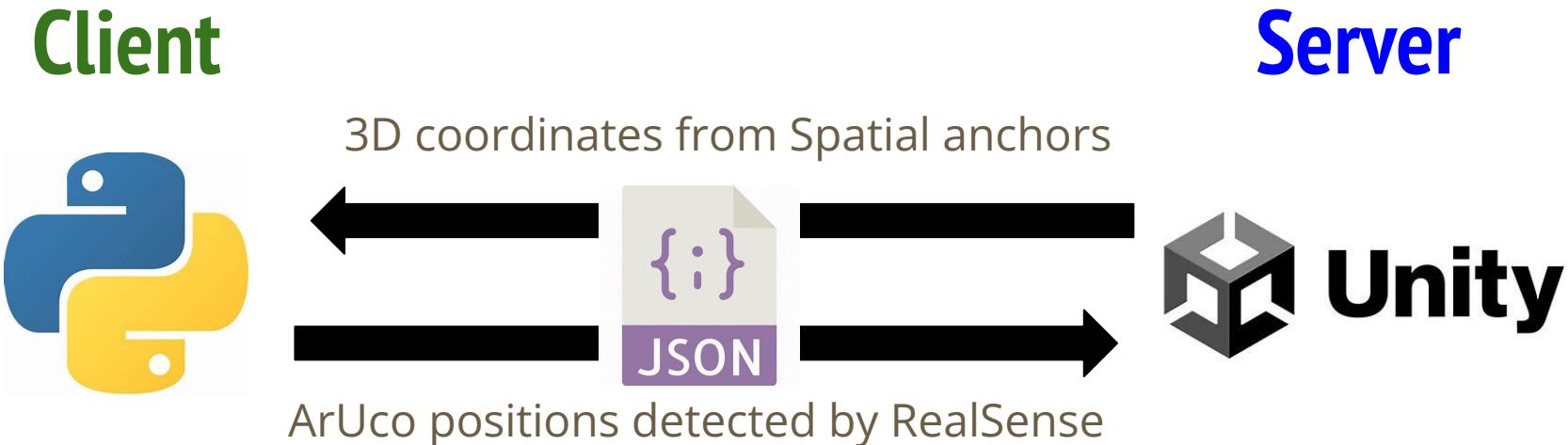
Solve n linear equations to find 3D Transformation matrix T.

# Spawn new objects



# Server Client

# Server Client



# Unity

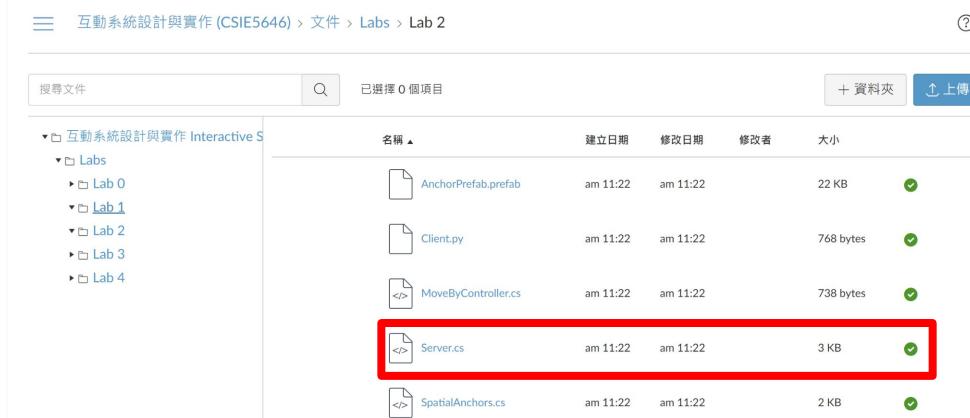
A simple server that send message to clients every two second.

```
17     const string hostIP = "127.0.0.1"; // Select your IP  
1 reference  
18     const int port = 80; // Select your port
```

三 互動系統設計與實作 (CSIE5646) > 文件 > Labs > Lab 2

搜尋文件  + 資料夾

| 名稱                  | 建立日期     | 修改日期     | 修改者 | 大小        |
|---------------------|----------|----------|-----|-----------|
| AnchorPrefab.prefab | am 11:22 | am 11:22 |     | 22 KB     |
| Client.py           | am 11:22 | am 11:22 |     | 768 bytes |
| MoveByController.cs | am 11:22 | am 11:22 |     | 738 bytes |
| Server.cs           | am 11:22 | am 11:22 |     | 3 KB      |
| SpatialAnchors.cs   | am 11:22 | am 11:22 |     | 2 KB      |



# Unity

```
// Receive message from client
int i;
while ((i = stream.Read(buffer, 0, buffer.Length)) != 0)
{
    data = Encoding.UTF8.GetString(buffer, 0, i);
    Message message = Decode(data);
    // Add received message to que
    lock(Lock)
    {
        MessageQue.Add(message);
    }
}
client.Close();
```

# Unity

```
private void Update()
{
    // Send message to client every 2 second
    if(Time.time > timer)
    {
        Message msg = new Message();
        msg.some_string = "From Server";
        msg.some_int = 1;
        msg.some_float = .1f;
        SendMessageToClient(msg);
        timer = Time.time + 2f;
    }
    // Process message que
    lock(Lock)
    {
        foreach (Message msg in MessageQue)
        {
            // Unity only allow main thread to modify GameObjects.
            // Spawn, Move, Rotate GameObjects here.
            Debug.Log("Received Str: " + message.some_string + " Int: " + message.some_int + " Float: " + message.some_float);
        }
        MessageQue.Clear();
    }
}
```

# Unity - To Do

- Send Spatial Anchor positions to Python
- Create Anchor from ArUco coordinates send by Python
  - Create all anchors from Python under the same parent.
  - Implement script to move the parent. (For manual calibration)
- Implement your own message format.
  - ArUco Id
  - x,y,z coordinates
  - Rotation
  - etc.

```
24     // Define your own message
25     [Serializable]
26     8 references
27     public class Message
28     {
29         2 references
30         public string some_string;
31         2 references
32         public int some_int;
33         2 references
34         public float some_float;
35     }
```

# Python

A simple client that echo message from server.

```
8     HOST = "127.0.0.1"  # The server's hostname or IP address
9     PORT = 80             # The port used by the server
```

三 互動系統設計與實作 (CSIE5646) > 文件 > Labs > Lab 2

搜尋文件  Q 已選擇 0 個項目 + 資料夾

| 名稱                  | 建立日期     | 修改日期     | 修改者 | 大小        |
|---------------------|----------|----------|-----|-----------|
| AnchorPrefab.prefab | am 11:22 | am 11:22 |     | 22 KB     |
| Client.py           | am 11:22 | am 11:22 |     | 768 bytes |
| MoveByController.cs | am 11:22 | am 11:22 |     | 738 bytes |
| Server.cs           | am 11:22 | am 11:22 |     | 3 KB      |
| SpatialAnchors.cs   | am 11:22 | am 11:22 |     | 2 KB      |

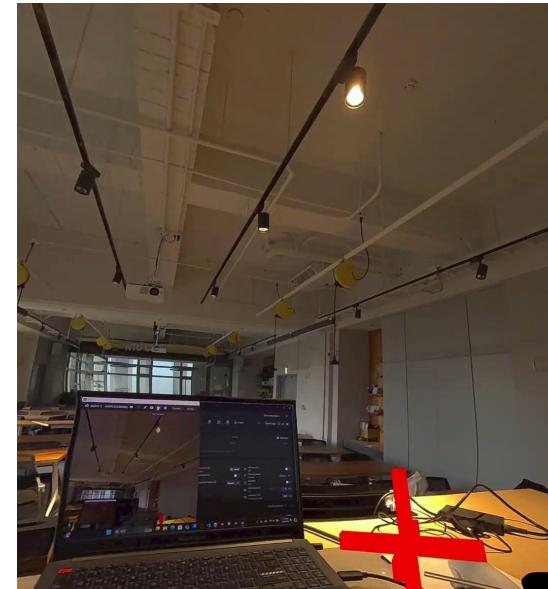
# Python - To Do

1. Receive Spatial Anchor positions from Unity.
2. Find the 3D transformation matrix that maps RealSense space to Unity Space.
  - a. Use numpy.linalg.lstsq to find the least square solution.
3. Use the 3D transformation matrix to transform ArUco position detected by RealSense to Unity space.
4. Send ArUco position detected by RealSense to Unity.

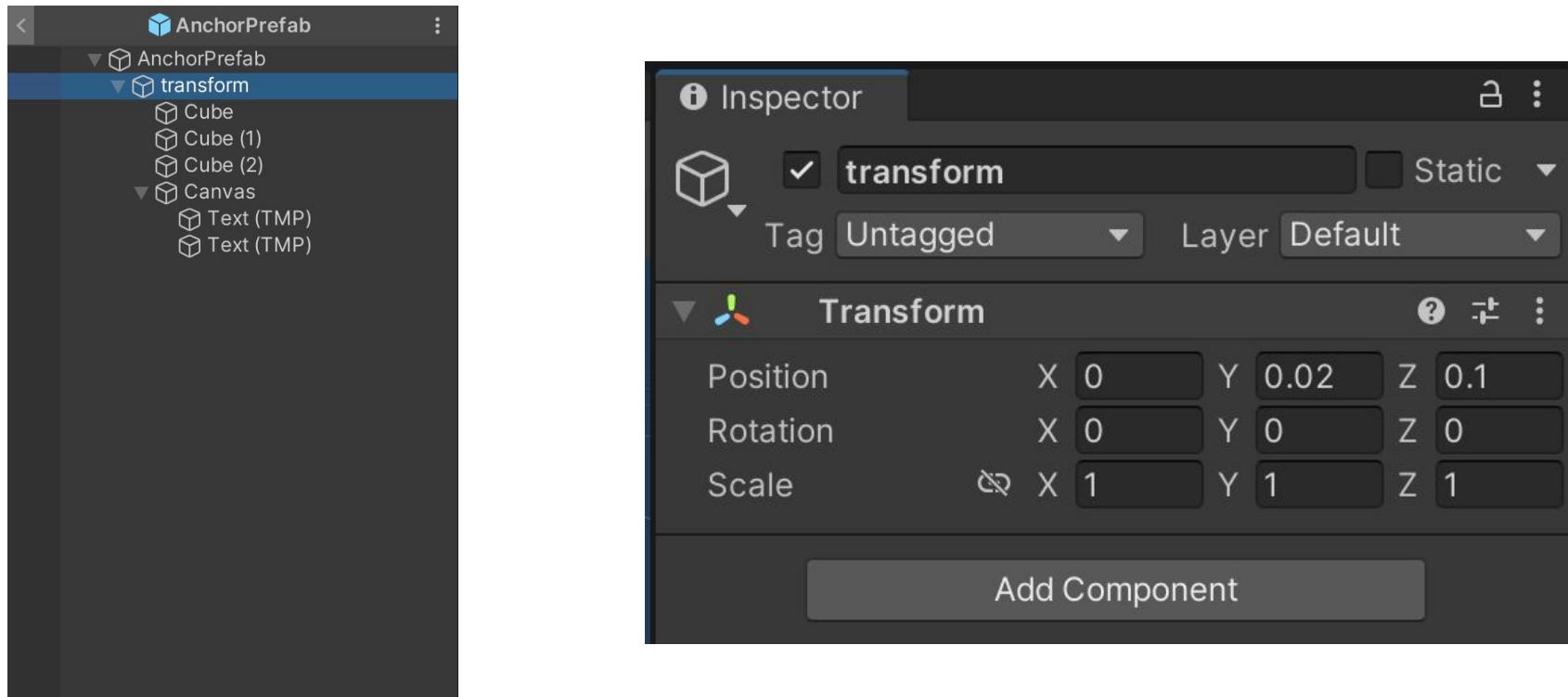
# To Do

# Basic - 1

- Modify the Transform in Anchor Prefab, and add physical objects to controller for user to create anchor at contact area.



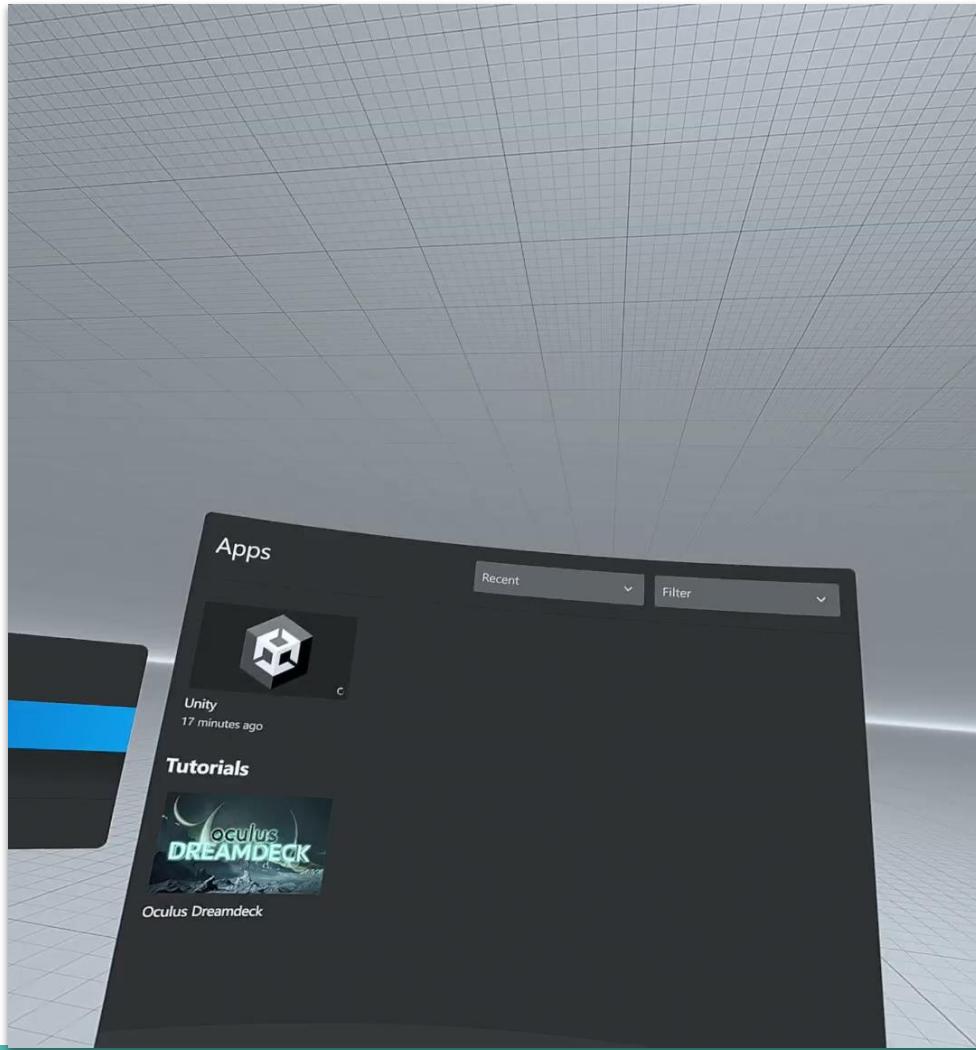
# Modify the Transform in Anchor Prefab



## Basic - 2

### Implement the calibration process.

- User can mark ArUco position with spatial anchors.
- User can see and adjust manually the anchors detected by RealSense in VR.



# Report

- Summarize what you did in this lab.
  - Show the basic and bonus you implemented.
  - Use screenshots & video demos to show your results.
- What you did to improve this device. Or how you can improve this device.
- Anything related to this lab.

# Hints

- Instantiate all objects from RealSense under the same parent.
- Add a script to move the parent object by Quest controller.
  - Every child will move with the parent.
  - Add rotation and movement on z-axis.

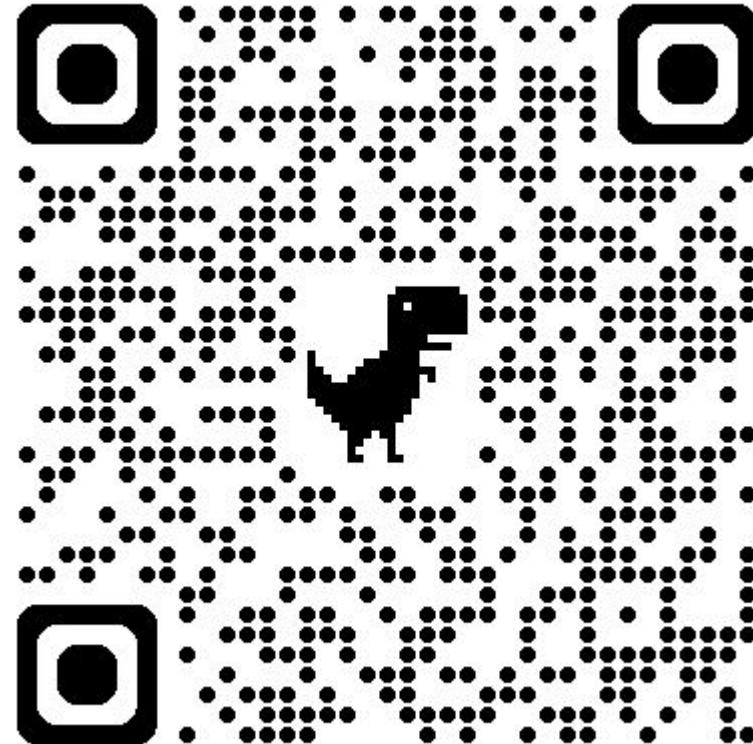
```
void Update()
{
    // Get value from controller thumb stick.
    Vector2 axis = OVRInput.Get(OVRInput.Axis2D.PrimaryThumbstick, controller);
    // Move the GameObject on xy plane.
    transform.Translate(new Vector3(axis.x, 0, axis.y) * speed * Time.deltaTime, Space.World);
}
```

# Hints

Server client message should support future applications.

- Spawn new objects in game according to the ArUcos IDs detected.
  - Is it a new one ? or the previous frame didn't detect it ?
- Track the movement and rotations of ArUco IDs.
  - How to distinguish between error and real movement ?
- Destroy game objects if the ArUco IDs are removed.
  - Is it removed ? or player accidentally blocked it ?

# Feedbacks



<https://forms.gle/RkRvD5nYEznGr2Jd7>