SeerLens™ One AR Glass

Application Development Manual

Xvisio Technology (Shanghai) Co., Ltd.

Contents

1. Setup Development Environment

It is recommend to use Unity2020.3.14f1c1. It is not recommended to use Unity2019.4.x LTS because of a memory leak problem.

Due to the differences between different unity versions, user can try to replace it with the recommended version if the un-recommended version encounters problems.

1.1 Setup Android Development Environment

SW Name	SW Version
JDK	JDK 1.8.0 and above
Android SDK	API Level 19 and above

Table 1-1 Setup Android Development Environment

1.2 Setup URP Development Environment

SW Name	SW Version
Unity	2020.3.14
URP	7.1.8

Table 1-2 Setup URP Development Environment

1.3 Development Considerations

1.3.1 Unity considerations

- FPS may different with different Unity version.
- In the Unity editor running mode, moving with Alt + mouse is equivalent to the effect of head rotation, moving with Ctrl + mouse is equivalent to the effect of head rotation around the Z axis.
- SDK provides a one-click XR configuration interface in the editor to convenient developers. The interface is located in the toolbar “Nibiru XR - XR settings”:

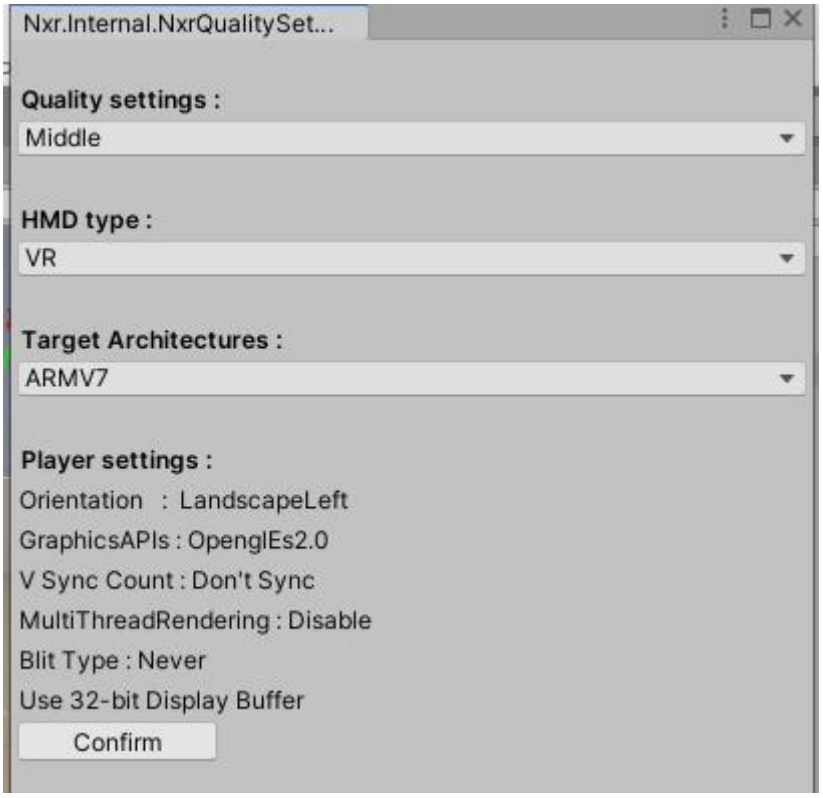


Figure 1-1 “Nibiru XR - XR settings”

“Quality settings” include Low/Middle/High.

- Low: Turn off anti aliasing
- Middle: 2X anti aliasing
- High： 4X anti aliasing

“HMD Type” indicates the target type of development (must option)

- VR : Applications developed for VR machines
- AR : Applications developed for AR machines

“Target Architectures” indicates the package is 32-bit or 64-bit. It is default to 32-bit.

Whether 64-bit works normally depends on the configuration of PC system.

“Player settings” is used to modify default packaging configuration parameters.

- Orientation: LandscapeLeft
- GraphicsAPIs:OpenglEs2.0
- Use 32-bit Display Buffer

The configuration is automatically modified after click “Confirm”.

1.3.2 Android Manifest File

If the file “AndroidManifest” doesn’t contain in the application, please use it in the SDK directly. If it is included in the application, developer needs to merge the “AndroidManifest”.

- Activity needs to inherit from com.nibiru.lib.xr.unity.NibiruXRUnityActivity.

Intent-filter needs to add：

```
<category android:name="com.google.intent.category.CARDBOARD" />
 <category android:name="com.nibiru.intent.category.NVR" />
<category android:name="com.nibiru.intent.category.STUDIO" />
```

- Plug-in manifest:

```
  <!-- "6DOF", "RECORD", "MARKER" !-->
  <meta-data android:value="6DOF " android:name="NIBIRU_PLUGIN_IDS"/>
 <meta-data android:value="NxrViewerMain"
android:name="NIBIRU_UNITY_VIEWER_NAME"/>
```

- Target device type:

```
 <!--0=vr,1=ar-->
 <meta-data android:value="VR" android:name="HMD_TYPE"/>
```

- Encrypted version or not：

```
<!—encrypted version-->
  <meta-data android:value="0"
android:name="NIBIRU_ENCRYPTION_MODE"/>
```

- Add necessary root:

```
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
  <uses-permission android:name="android.permission.BLUETOOTH"/>
  <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
  <uses-permission
```

```
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
   <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />
   <uses-permission android:name="android.permission.INTERNET" />
   <uses-permission android:name="android.permission.GET_TASKS" />
   <uses-permission android:name="android.permission.CAMERA"/>
   <uses-permission
android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS"/>
   <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"/>
   <uses-permission android:name="android.permission.WRITE_SETTINGS"/>
 <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
```

1.3.3 SDK Recommended Configuration

- Graphics APIS doesn't support Vulkan. Developers need to choose OpenGLES2 or OpenGLES3 from their own requirements.
- "MultiThreaded Rendering" is not supported. No need to select.
- Select "Landscape Left" of "Default Orientation".
- Select "Never" of Blit Type.

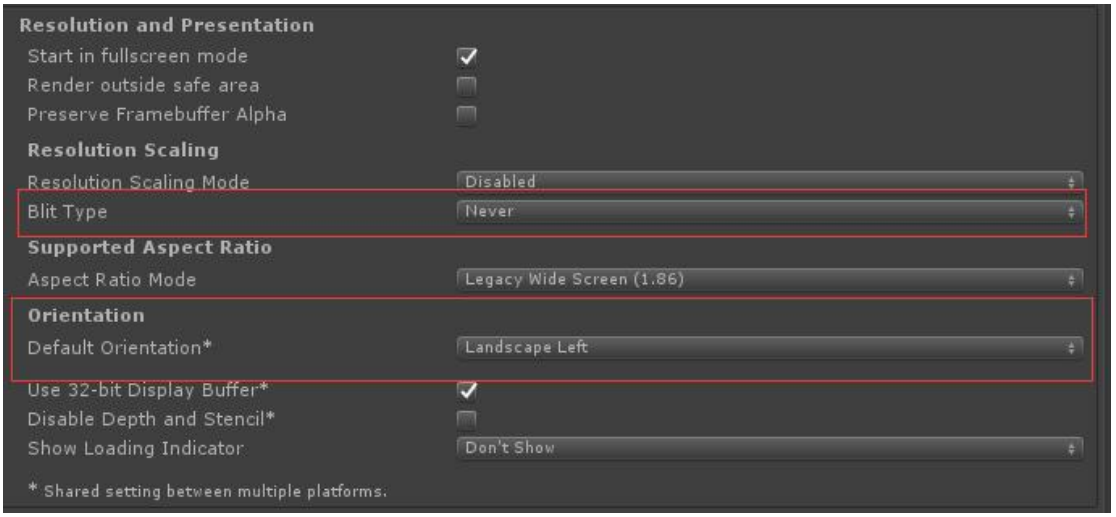


Figure 1-2 Setting

- Select “Don’t Vsync” of “V Sync Count” in “Quality”.

Figure 1-3 Quality

- Edit “/Project Settings/Graphics”，add “NXR/Resource/UnlitTexture.shader” and “NXR/Resource/SolidColor.shader” into list “Always Included “Shaders”.

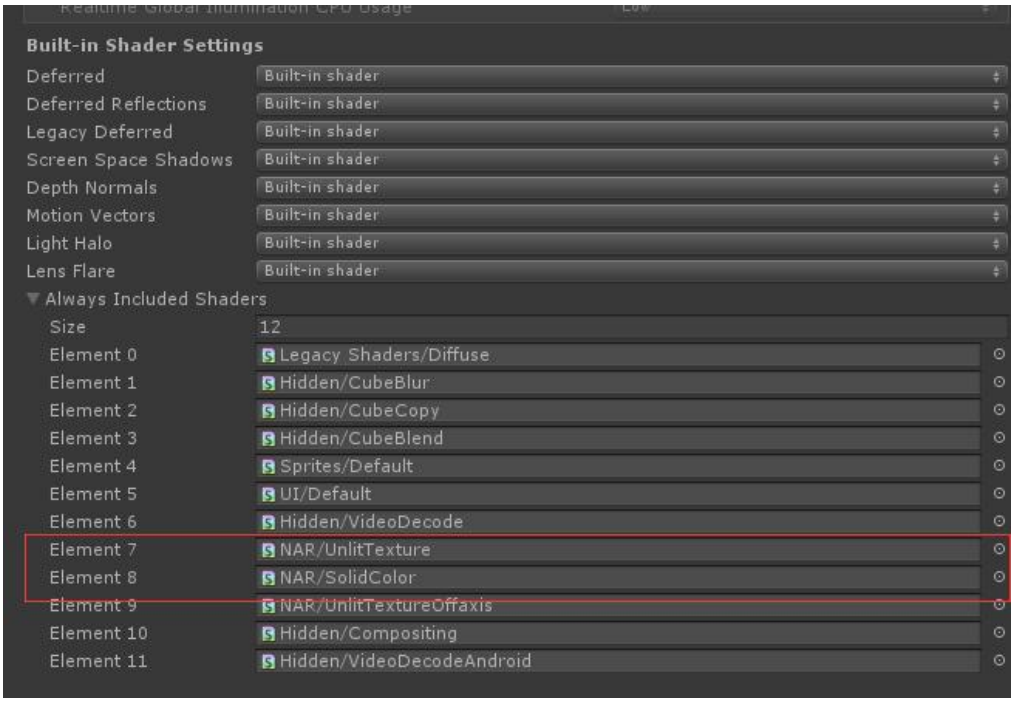


Figure 1-4 Bulit-in Shader Settings

1.3.4 Configuration of Universal RP

- Unity version：**Unity2020.3.14**
- URP version: **V7.1.8**

If the project uses URP, please refer to the current configuration description for configuration adaptation.

- Open “Window/Package Manager” to search “Universal RP” to install.

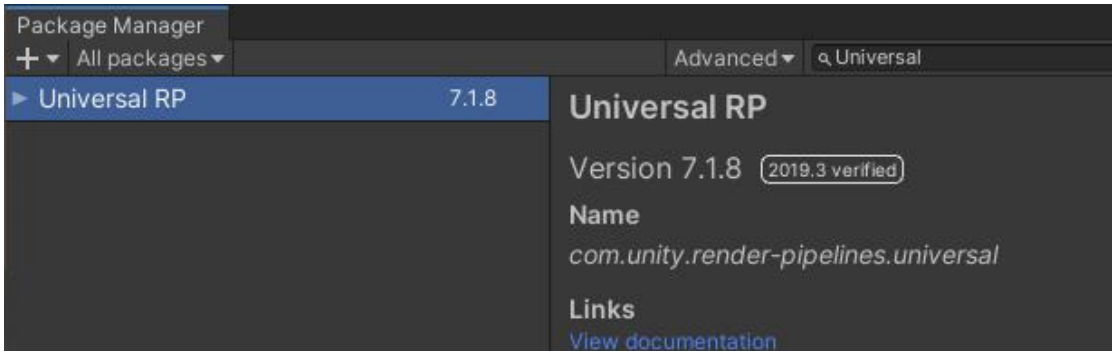


Figure 1-5 Package Manager

- Installation completed. Create “UniversalRenderPipelineAsset , Assets/Create/Remdering/Universal Render Pipeline/Pipeline Asset”.
- Open “Edit/Project Settings/Graphics/”, select “PipelineAsset” in “Scriptable Render Pipeline Settings”.

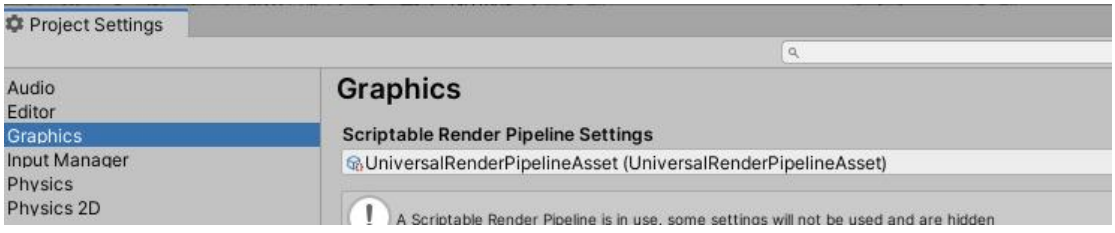


Figure 1-6 “Project Settings”

- Open “Edit/Project Settings/Quality” and select Android category. Select “PipelineAsset” in “Rendering”.

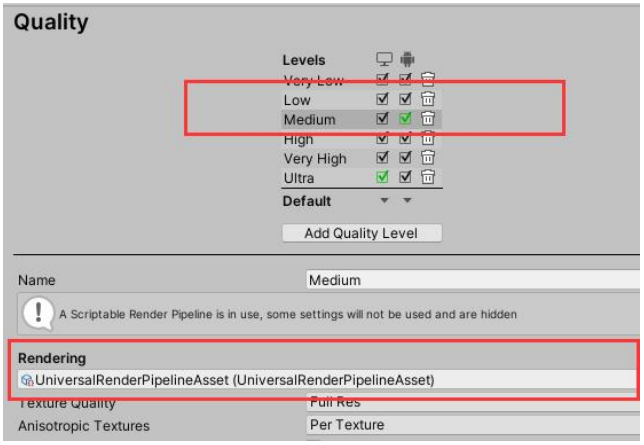


Figure 1-7 “Quality”

- Upgrade engineering materials:

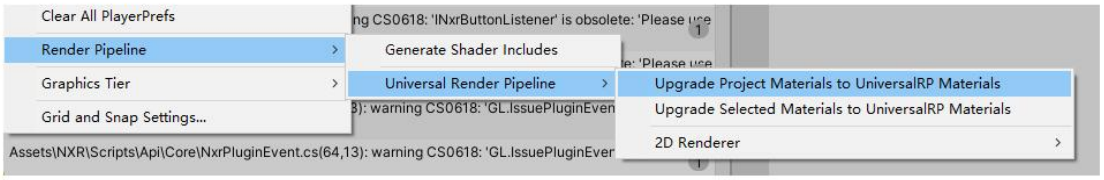


Figure 1-8 “Upgrade Engineering Materials”

Note: Don’t select “SRP Batcher/Dynamic Batching” in “UniversalRenderPipelineAsset”. Otherwise, memory leakage will occur.

Figure 1-9 Advanced

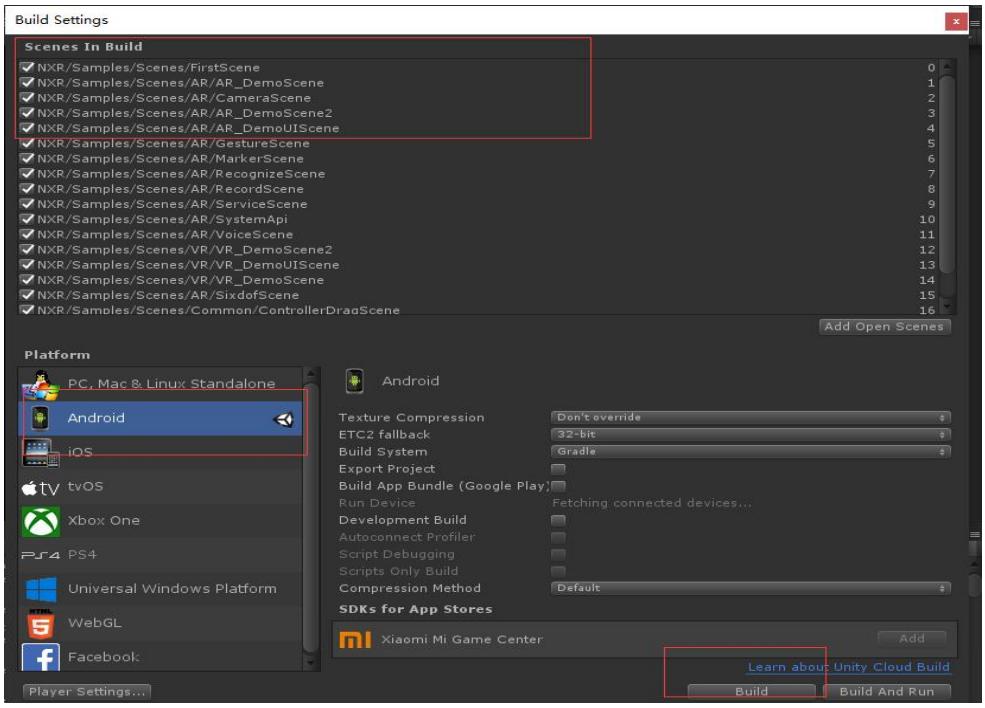


Figure 1-10 Build Settings

2. BOX Information

2.1 Update

① Open setting:

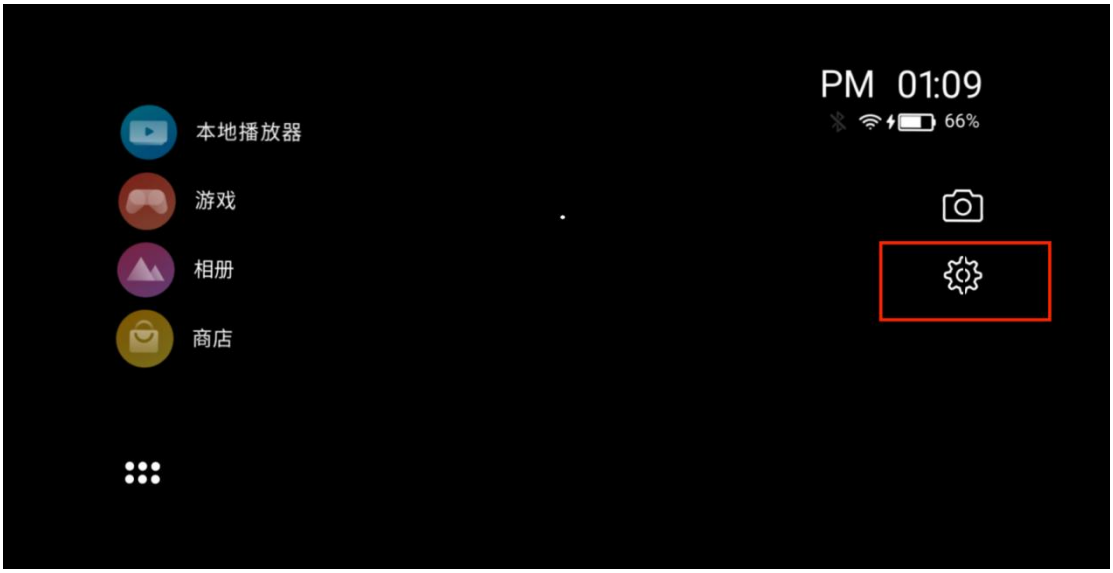


Figure 2-1 System

② Select “system” and then select “system update”:

Figure 2-2 System Update

③ Select “upgrading online”:

Figure 2-3 Upgrading Online

④ Click “OK” after the upgrade package is detected

Figure 2-4 Upgrade Confirmation

⑤ After confirmation, the system will download the upgrade package automatically.

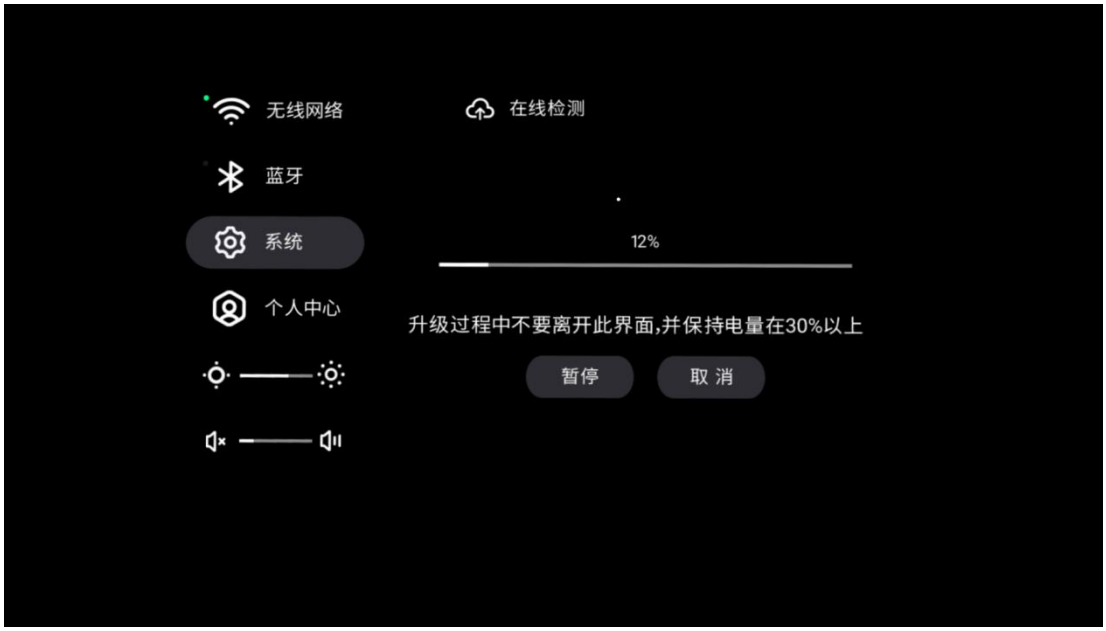


Figure 2-5 Upgrading

⑥ Click “update now” after download completed.

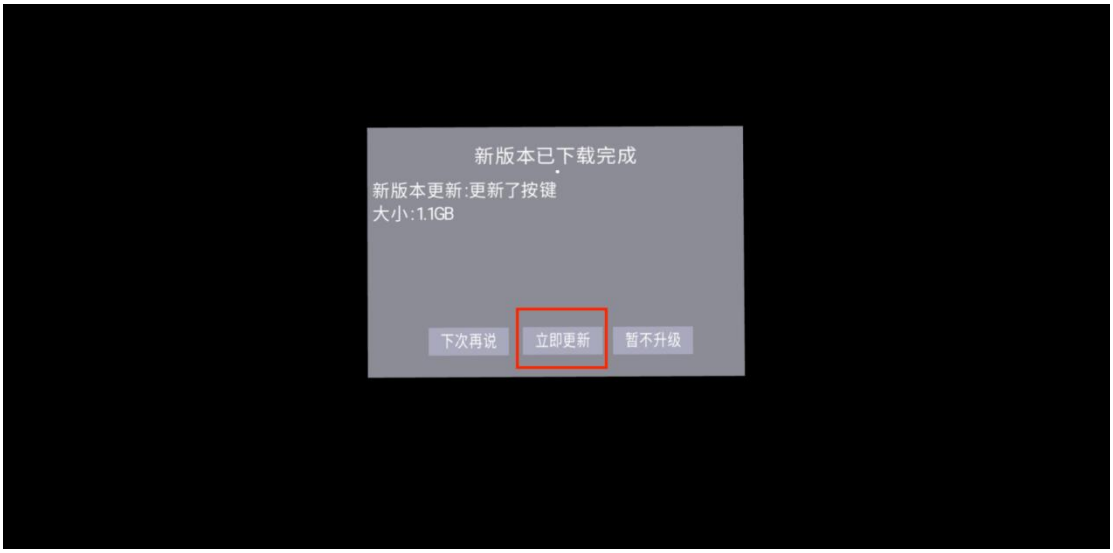


Figure 2-6 Update System

⑦ System will restart automatically after the update is completed which indicates the system upgrade is completed.

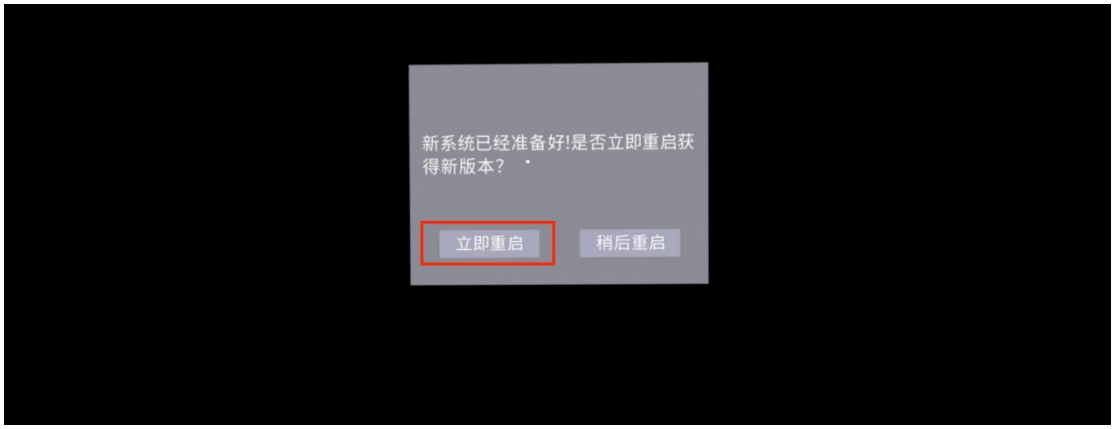


Figure 2-7 Restart System

3. SDK Application

3.1 Use anchor point to do selection

【example scene：NXR/Samples/Scenes/FirstScene】

Function	Function name and interface
System white point	NxrViewer.Instance.HeadControl = HeadControl.GazeSystem;
APP white point	NxrViewer.Instance.HeadControl = HeadControl.GazeApplication;

APP white point Hover	NxrViewer.Instance.HeadControl = HeadControl.GazeHover;

Table 3-1 Functions 1

Function	**Function name and interface**
Adjust the size of system white point	NxrViewer.Instance.GazeApi(GazeTag.Set_Size, ((int) GazeSize.Larger).ToString());
Adjust the color of system white point	NxrViewer.Instance.GazeApi(GazeTag.Set_Color, "0.043f_0.435f_0.043f"); RGB range（0~1）
Adjust visibility of system white point	NxrViewer.Instance.GazeApi(GazeTag.Hide);

Table 3-2 Function 2

The main difference between the system white dot and the app white dot is that the app white dot will shake due to the influence of frame insertion, the system white dot does not have this problem.

Objects such as cube in the scene can be targeted, but how can we monitor the targeting event? SDK provides an interface “INxrGazeResponder.cs” which can be used to achieve the function of aiming entry, exit, confirmation and other events.

We provide an example “Teleport.cs” in Demo, example, attach it to the cube object, add the component “Event Trigger” to the cube. Add the “Pointer Enter”, “Pointer Exit” and “Pointer Click” under the “Event Trigger”.

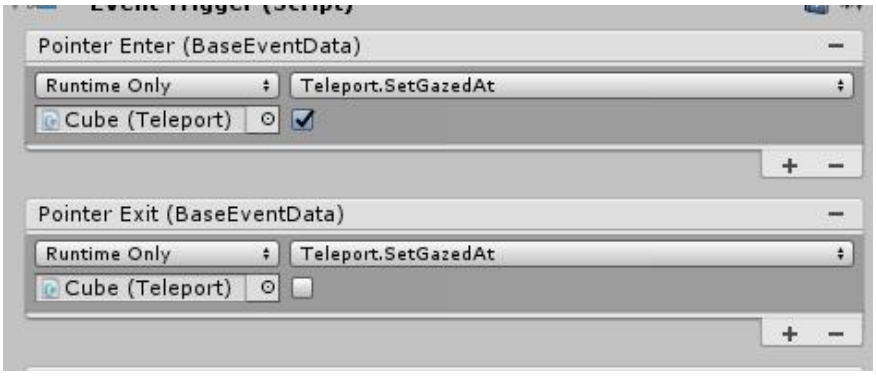


Figure 3-1 Setting

Figure 3-2 Cube

It will change color if the cube above is selected. After the cube is selected, pressing the “OK” key will trigger the cube to change its position randomly.

For the “Button” component click event in Canvas, you can directly add processing logic to the corresponding “On Click” event.

3.2 Interfaces of view lock and view reset

Function	Function name and interface
View lock	NxrViewer.Instance.LockHeadTracker=true;
View reset	NxrViewer.Instance.Recenter();

Table 3-3 View Lock and View Reset

3.3 Texture Quality

Option “Texture Quality” is behind “NxrViewer”. It is default to “Good”(image is normal and clear), the other two options are Simple（worse and dim image）and Best （the best and clearest image）.

Note: Modify in “NXR/Prefabs/NxrViewerMain”. Modifying in a separate scene is appropriate for the editor to run.

3.4 Related interfaces of system

Function	Function name and interfaces
System split screen: adapt to whole scene 2D UI. Call the system split screen now instead of using our SDK split screen.	NxrViewer.Instance.SetSystemSplitMode(flag); （flag=1 the screen is divided into left and right screen, flag=0 Application of split screen indicates the split screen effect produced by mounting our camera script

Get Android SD card path. Return path, for example: /storage/emulated/0	NxrViewer.Instance.GetStoragePath();
The virtual mouse registration service is only applicable to 2D applications in normal split screen mode.	NxrViewer.Instance.GetNibiruService(). RegisterVirtualMouseService();
Virtual mouse logoff service	NxrViewer.Instance.GetNibiruService(). UnRegisterVirtualMouseService();
Virtual mouse control	NxrViewer.Instance.GetNibiruService(). SetEnableVirtualMouse(bool enabled);
Get network status: 0=disable network，1=enable network	NibiruTaskApi.GetNetworkStatus(); NibiruTaskApi.IsWifiEnabled();
Get status of Bluetooth: 0=disable，1=enable	NibiruTaskApi.GetBluetoothStatus(); NibiruTaskApi.IsBluetoothEnabled();
Get MAC address	NibiruTaskApi.GetMacAddress();
Get device unique identification	NibiruTaskApi.GetDeviceId();
Start apk	NibiruTaskApi.LaunchAppByPkgName("com.nibiru.vr.lib2.test");
Enable/disable WiFi	NibiruTaskApi.SetWifiEnable(bool enable);
Enable/disable bluetooth	NibiruTaskApi.SetBluetoothEnable(bool enable);
Install apk	NibiruTaskApi.InstallApk(string apkPath);
Install apk success/failed callback	NibiruTaskApi.SetInstallSuccessCallback(InstallSuccessCallback callback); NibiruTaskApi.SetInstallFailedCallback(InstallFailedCallback

	callback);
Install APK to start automatically	NibiruTaskApi.InstallAndStartApk(string apkPath);
Uninstall apk	NibiruTaskApi.UninstallApk(string packageName);
Uninstall apk success/failed callback	NibiruTaskApi.SetUninstallSuccessCallback(UninstallSuccessCallback callback); NibiruTaskApi.SetUninstallFailedCallback(UninstallFailedCallback callback);

Table 3-4 Interfaces of Functions

NibiruService nibiruService = NxrViewer.Instance.GetNibiruService();

Function	**Function name and interface**
Get vendor software version	nibiruService.GetVendorSWVersion();
Get light sensor value: (do not call frequently)	nibiruService.GetLightValue();
Get distance sensor value: (do not call frequently)	nibiruService.GetProximityValue();
Get screen brightness value:	nibiruService.GetBrightnessValue();
Adjust screen brightness	nibiruService.SetBrightnessValue(nibiruService.GetBrightnessValue() - 1);
Disable/enable touchpad cursor	nibiruService.SetEnableTouchCursor(false);
Get device unique identification	NibiruTaskApi.GetDeviceId();
Start apk	NibiruTaskApi.LaunchAppByPkgName("com.nibiru.vr.lib2.test");

Table 3-5 Interfaces of Functions

3.5 6DOF Displacement

【example scene：NXR/Samples/Scenes/MainScenes/SixdofScene】

NXR/Samples/Scripts/MainScenes/SixdofTest

Figure 3-3 NXR Viewer

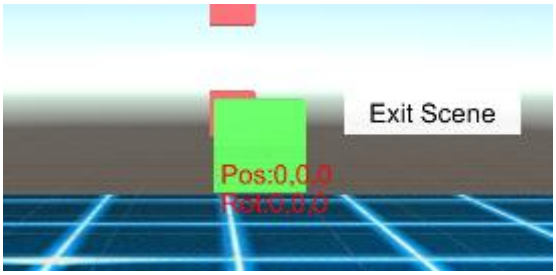


Figure 3-4 Location

Pos: 0,0,0 indicated the location (x,y,z) information.

Option "Tracker Position" behind "General Settings" in script "NxrViewerMainO" indicates enable/disable displacement function. Call "NxrViewer.Instance.TrackerPosition=false" or modify "NXR/Prefabs/NxrViewerMain" to disable the option.

Function	**Function name and interfaces**
Monitor the change of displacement information	NxrViewer.onSixDofPosition += OnHeadPosition; void OnHeadPosition(float x, float y, float z) NxrViewer.onSixDofPosition -= OnHeadPosition;

Table 3-6 Interfaces of Functions

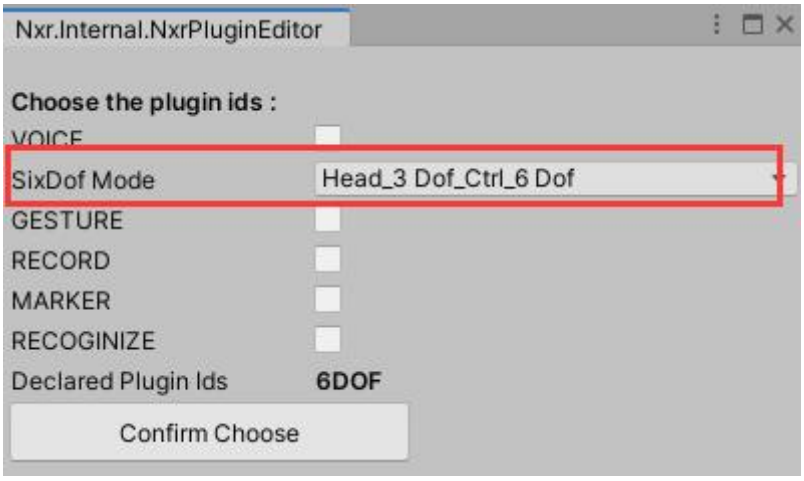


Figure 3-5 Set 6DOF Mode

For 6DOF, three modes are provided in the plug-in selection interface: head 3DOF

handle 3DOF, head 3DOF handle 6DOF (default selection) and head 6DOF handle 6DOF.

Note: All the displacements of 6DOF are absolute.

3.6 Permission Application

Function	Function name and interfaces
Permission application	NxrViewer.Instance.GetNibiruService().RequsetPermission(string[])

Table 3-7 Interfaces of Functions

Permission name list:

Camera: "android.permission.CAMERA";

Abbreviation of external storage:

"android.permission.WRITE_EXTERNAL_STORAGE";

Read external storage: "android.permission.READ_EXTERNAL_STORAGE";

Location: "android.permission.ACCESS_COARSE_LOCATION";

Network state: "android.permission.ACCESS_NETWORK_STATE";

Setting: "android.permission.WRITE_SETTINGS";

Bluetooth: "android.permission.BLUETOOTH";

Bluetooth admin: "android.permission.BLUETOOTH_ADMIN";

Internet: "android.permission.INTERNET";

Tasks: "android.permission.GET_TASKS";

Record: "android.permission.RECORD_AUDIO";

Phone state: "android.permission.READ_PHONE_STATE";

If the permission cannot be applied or the application fails, user can modify “Player Settings/Other Settings/Target API Level” and select API level 22 to bypass the permission problem.

SDK will output a prompt log if the current system does not support it:

*******[RequestPermission Warnning]*******

*******System Currently Not Support, Please Take Care !!!******* ……

3.7 SLAM User Guide

3.7.1 Introduction of SLAM Example Scenes

【example scene: NXRSlam/Demo/SlamScene】

The test scenario includes the demonstration of functions related to map, camera and plane detection. Switch the test page through the following buttons.

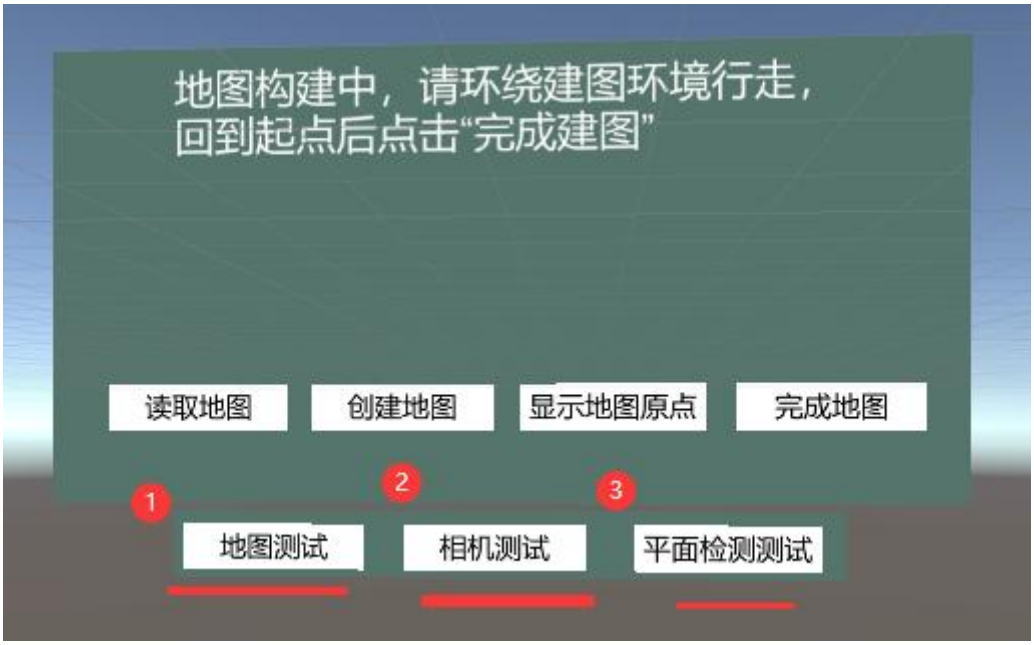


Figure 3-6 SLAM

3.7.2 SLAM Interface Function

NxrSlam.InstanceFunction name	**Parameter**	**Return value**	**Description**
Init	nil	bool true init success false init failed	Initial
EnableNativeLog	bool enable		Enable/disable native log print
MapStatusAndQualityEvent	int map quality		Monitor map state

			changes and quality event
MapPercentEvent	float percent of map coincidence		Monitor map coincidence
StreamDataEvent	Support stream data callback of RGB , TOF , FISHEYE		Monitor camera data callback
PlaneDetectionEvent	Support data callback of plane points and normal vectors		Monitor data callback of plane detection
UnInit	Nil		Log off
StartPlaneDetection	Nil	bool true success false failed	Start plane detection
StopPlaneDetection	Nil	bool true success false failed	Stop plane detection
StopBuildMap	String path of map	bool true success false failed	stop mapping
StopSlam	Nil	Nil	Stop Slam
StartSteaming	NxrSlam.RawDataType type	Nil	Start camera steam

	Camera type		
StopStreaming	NxrSlam.RawDataType type Camera type	Nil	Close camera stream
StartSlamWithMap	String path of map	bool true success false failed	Load the map which correspond to the path

Table 3-8 SLAM Interfaces

Demo example script:

NXRSlam/Demo/SlamApi.cs

All the interface call examples are included.

Interface call process:

Initial -> register callback monitor -> start camera, map and plane detection-> deal with callback data -> stop camera, map and plane detection -> log out

3.8 Introduction of Gesture Function

1) Distribution map of 25 feature points of gesture:

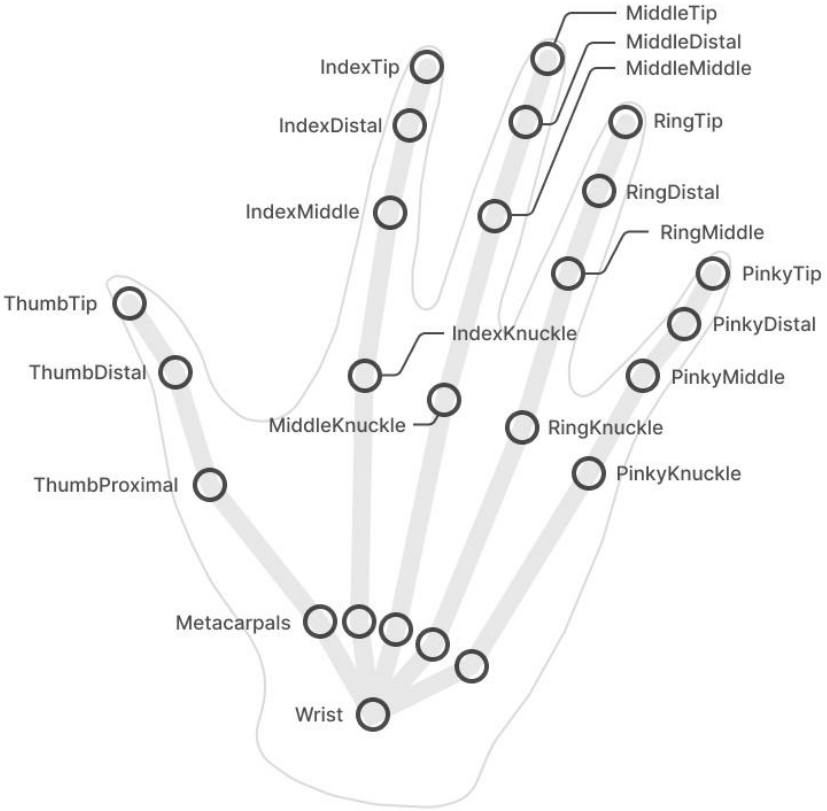


Figure 3-7 Gesture Distribution Map

2) Static gesture code：

	ID	Diagram
UNKONOW	-1	
GRAB	0	

Point	1	
VICTORY	2	
Three	3	
Four	4	
OpenHand	5	
Call	6	

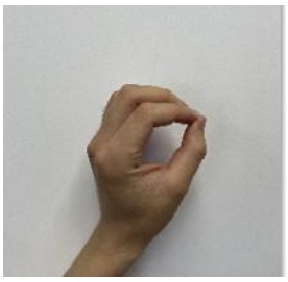
Rock	7	
Thumb	8	
Pinch	9	

Table 3-9 Static Gesture Code

3) Gesture interfaces:

```
public static extern int start_gesture_ex(xslam_skeleton_callback cb,int type);

public delegate void xslam_skeleton_callback(XvXRSkeleton skeleton);

public struct XvXRSkeleton{
        public int size;
        public Point[] joints_ex;
        public RotatePoint[] rotateData;
        public int[] guestureID;
        public long dataFetchTimeMs;
        public long dataTimeStampMs;
};
```

4. Unity APK

4.1 Deep Sea Demo

The demo mainly shows SLAM aiming and 3D gesture functions. The APK scene is a large immersive deep-sea world. User can interact with deep-sea fish through gesture touch. There are a variety of underwater creatures in the scene, including sharks, dolphins, etc. The fish will escaping after touching.

Figure 4-1 Deep Sea Demo

Details:

The following figure is the basic gesture component in the demo scene, which including the position information of 25 joint points of the left and right hands.

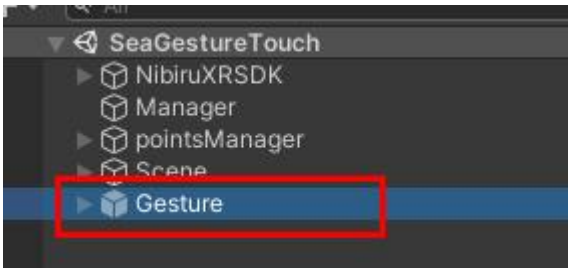


Figure 4-2 Gesture

1) Save the data of 25 gesture key points of the left and right hands into “Transform”, and start the gesture recognition function.

```
void start()
{
        for (int i = 0; i < 25; i++)
        {
```

```
            hand21List[i] = (lefthand21.GetChild(i).gameObject);
        }
        for (int i = 0; i < 25; i++)
        {
            hand21List[i + 25] = (righthand21.GetChild(i).gameObject);
        }
        GestreAction();
}
```

2) Enable the gesture recognition function. Execute the native code and the callback function, and process the data in the callback function

```
void GestreAction()
{
        Debug.Log("Gesture class start load ");
         try
         {
             AndroidJavaObjectjo=new
AndroidJavaObject("com.nibiru.ar.Gesture");
             float result;
             result = jo.Call<float>("getFloat", new object[] { });
             bool res = xslam_xrmoudle_open();
             start_gesture_ex(OnStartSkeletonCallback,1);
         }
         catch (AndroidJavaException e)
         {
             Debug.Log("Gesture class    load error ");
         }
}
```

3) Analysis of gesture data callback function

```
[MonoPInvokeCallback(typeof(xslam_skeleton_callback))]
```

```
public static void OnStartSkeletonCallback(XvXRSkeleton skeleton)
{
    //location
    for (int i = 0; i < joints_ex.Length; i++)
    {
        if (Double.NaN == skeleton.joints_ex[i].x || Double.NaN == skeleton.joints_ex[i].y || Double.NaN == skeleton.joints_ex[i].z)
        {
            joints_ex[i] = defultPoint;
        }
        else
        {
            if (skeleton.joints_ex[i].x == 0 && skeleton.joints_ex[i].y == 0 && skeleton.joints_ex[i].z == 0)
            {
                joints_ex[i] = defultPoint;
            }
            else
            {
            //Get the returned position of each gesture joint point and save it in the array.
                joints_ex[i] = new Vector3(skeleton.joints_ex[i].x, -skeleton.joints_ex[i].y, skeleton.joints_ex[i].z);
            }
        }
    }
  //Static gesture recognition ID acquisition
    GestureLeftID = int.Parse(skeleton.guestureID[0].ToString());
    GestureRightID = int.Parse(skeleton.guestureID[1].ToString());
```

```
            //Obtain the rotation angle of each gesture joint point.
            for (int i = 0; i < rotation_ex.Length; i++)
            {
                    RotatePoint rP = new RotatePoint();
                    rP.x = skeleton.rotateData[i].x;
                    rP.y = skeleton.rotateData[i].y;
                    rP.z = skeleton.rotateData[i].z;
                    rP.w = skeleton.rotateData[i].w;
                    rotation_ex[i] = rP;
                }
            }
}
```

4) Update the positions of 25 joint points, and display the static gesture ID.

```
void Update()
    {
        for (int i = 0; i < 50; i++)
        {
            if (double.IsNaN(joints_ex[i].x) == true)
            {
                continue;
            }
            hand21List[i].transform.position = joints_ex[i];
            Quaternion q = new Quaternion();
            q.x = -rotation_ex[i].x;
            q.y = rotation_ex[i].y;
            q.z = -rotation_ex[i].z;
            q.w = rotation_ex[i].w;
            hand21List[i].transform.rotation = q;
        }
```

```
        if (infoTxt != null)
            infoTxt.text = GestureLeftID + "===" + GestureRightID;
        if (idTxt != null)
            idTxt.text = GestureLeftID + "===" + GestureRightID;
}
```

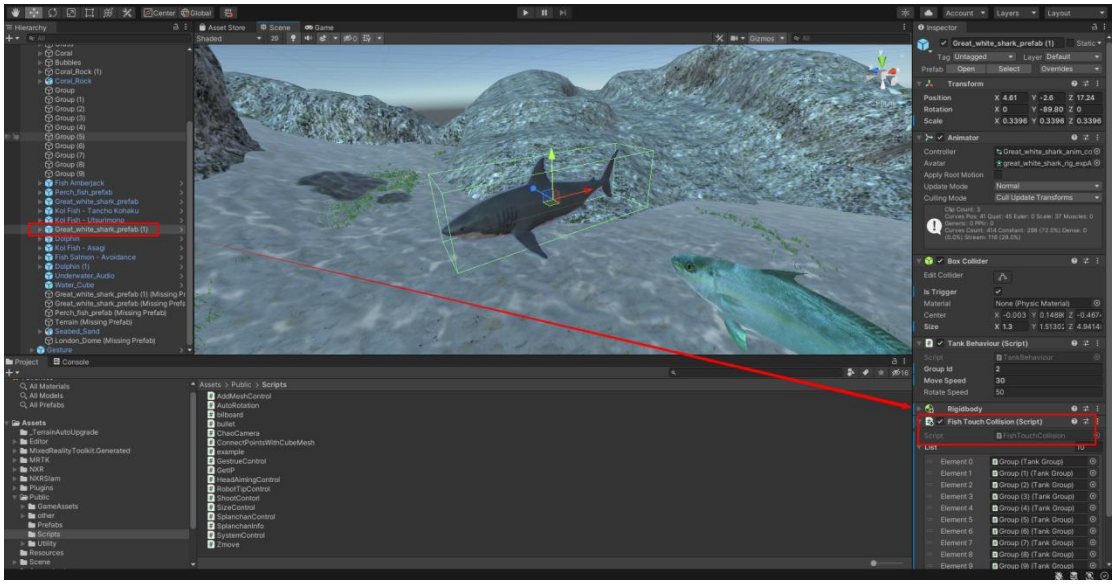


Figure 4-3 Details of “Deep sea” APK

5) The shark touch code is in “FishTouchCollision.cs”:

```
void OnTriggerEnter(Collider collider)
    {
        if (collider.name == "Cube (7)"){
list[this.gameObject.GetComponent<TankBehaviour>().groupId].transform.localPosit
ion = new Vector3(Random.Range(-1,4), Random.Range(-1, -1), Random.Range(11,
15));
        }
}
```

The shark will relocate to the new position once touched.

4.2 Robot Anchor Demo

The demo mainly shows the SLAM space aiming function. A robot will be pre-anchored in the middle of the scene。

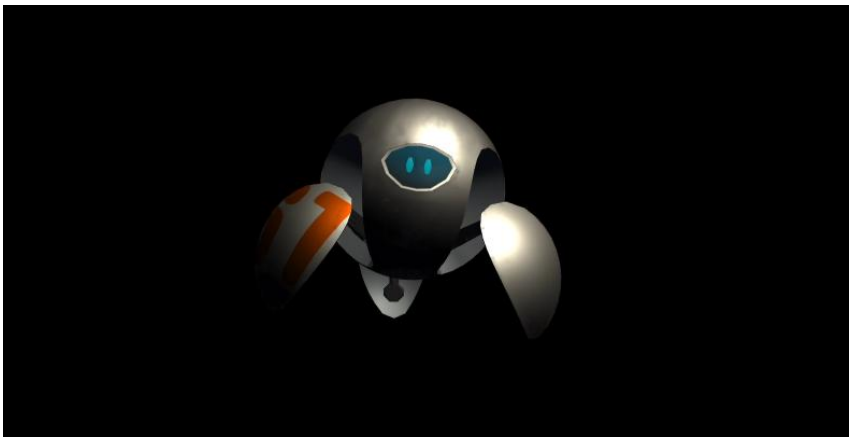


Figure 4-4 Robot Anchor Demo

The robot is anchored in the real scene when observe through the AR glasses. The robot can be displayed in the center of the viewing by press button “OK” in the box.

Detail:

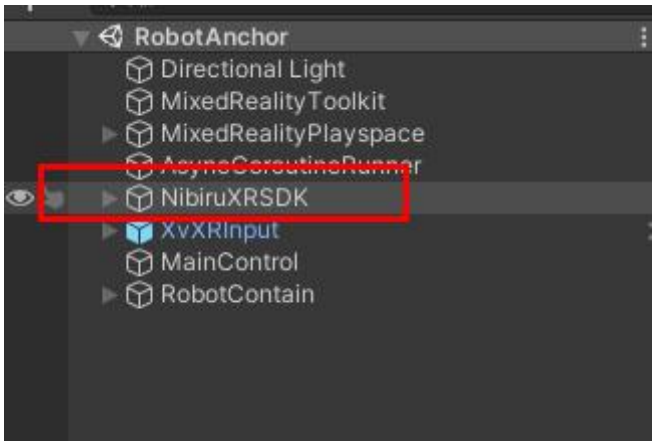


Figure 4-5 Nibiru SDK

“NibiruXRSDK” is the position of the head camera in the scene, including the settings of the left and right eye cameras and the head aiming. Put the position of “NibiruXRSDK” into (0,0,0) of Unity scene.

Monitor “ButtonUp” of “Fire1” in Update() to correspond with the operation of pressing button “OK” in BOX.

```
void Update()
    {
        //Press volume “↑” button to generate and anchor objects
        if (Input.GetButtonUp("Fire1") || Input.GetKey(KeyCode.F1))
```

```
        {
            for (int i = 0; i < rootContain.transform.childCount; i++)
            {
                Destroy(rootContain.transform.GetChild(i).gameObject);
            }

            robot = GameObject.Instantiate(robotPrefab, robotPrefab.transform.position,
robotPrefab.transform.rotation, transform);
            robot.transform.parent = rootContain.transform;
            robot.transform.position = robotPrefab.transform.position + new Vector3(0.069f, 0,
0);
            robot.SetActive(true);
        }
}
```

4.3 Gesture Demo

This demo mainly shows 3D gesture and SLAM function. User can catch or move the objects through gesture.

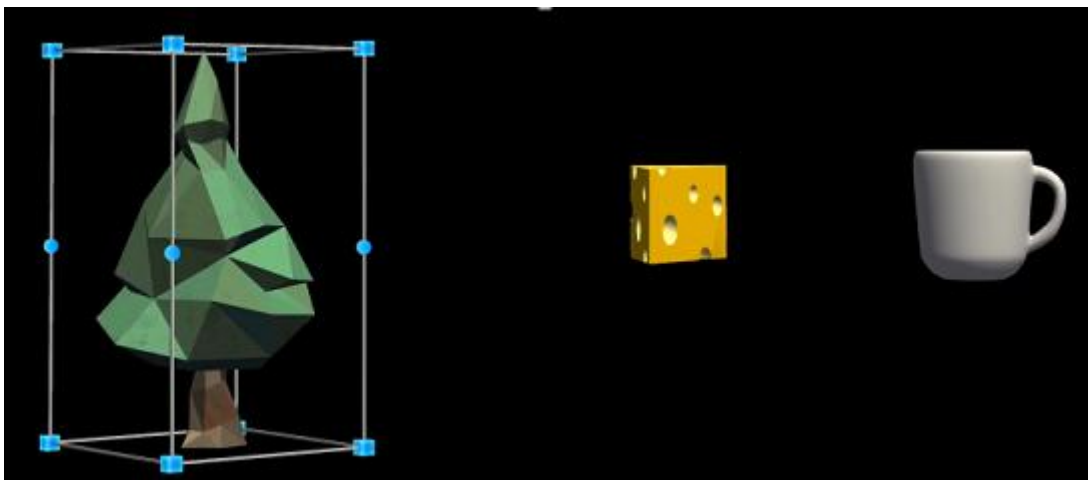


Figure 4-6 Objects

Refer to project demo scenario “Gesture.unity” for instance code. Refer to section 3.8 for gesture interfaces.

Method of replacing gesture joint point model:

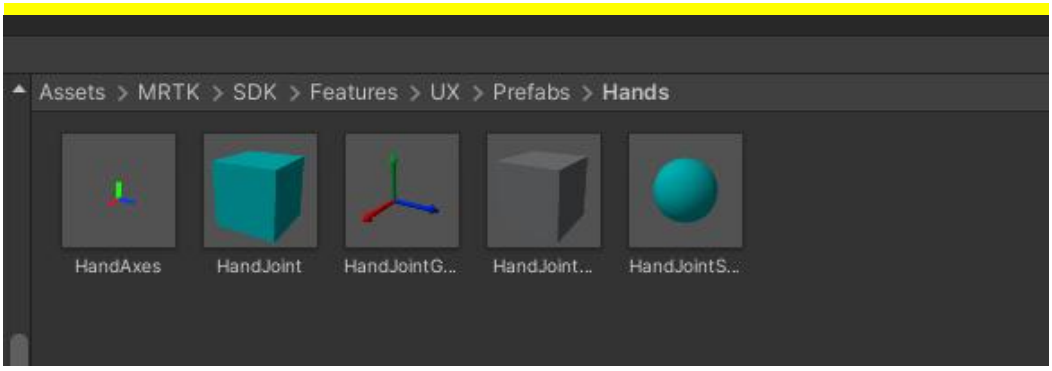


Figure 4-7 Model

5. APK Package Steps

1) Use verify tool to build new projects:

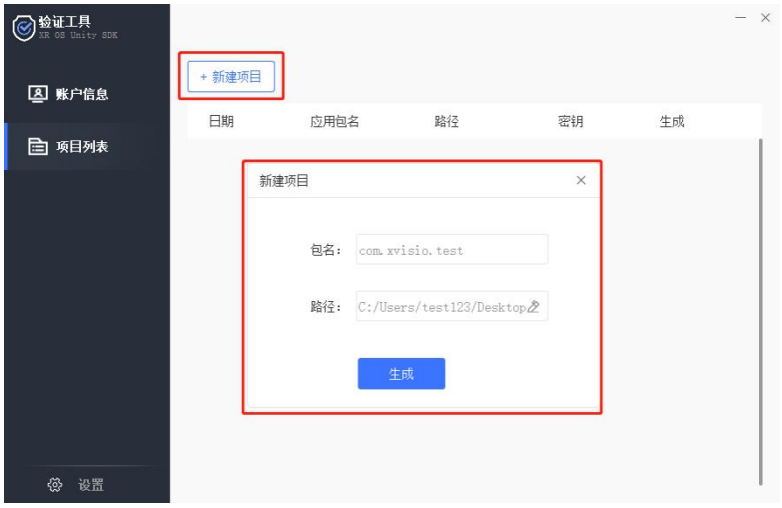


Figure 5-1 Build New Project

2) The package name is consistent with the “Package Name” in the “unit PlayerSettings”. Select directory “Assets\Plugins\Android\assets” in “unit” project as the path. The key of the project can be seen after generating.

Figure 5-2 Select Path

3) Click “NibiruXR => XR Settings” to set parameters and confirm.

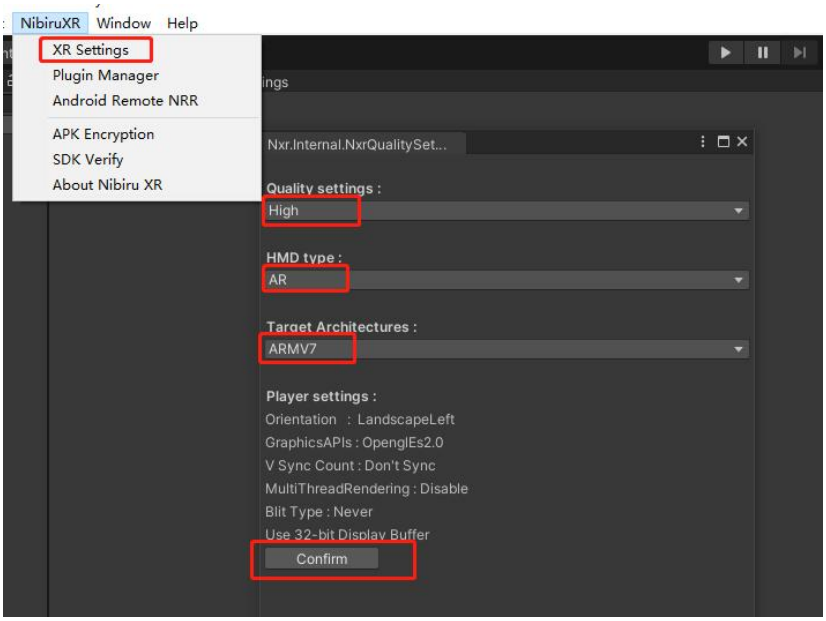


Figure 5-3 XR Setting

4) Click “NibiruXR => Plugin Manager”

“SixDof Mode”：6dof mode, select head control 6dof or 3dof.

“RECORD”：enable/disable record function.

“MARKER”：enable/disable marker function.

Click “Confirm Choose” to finish.

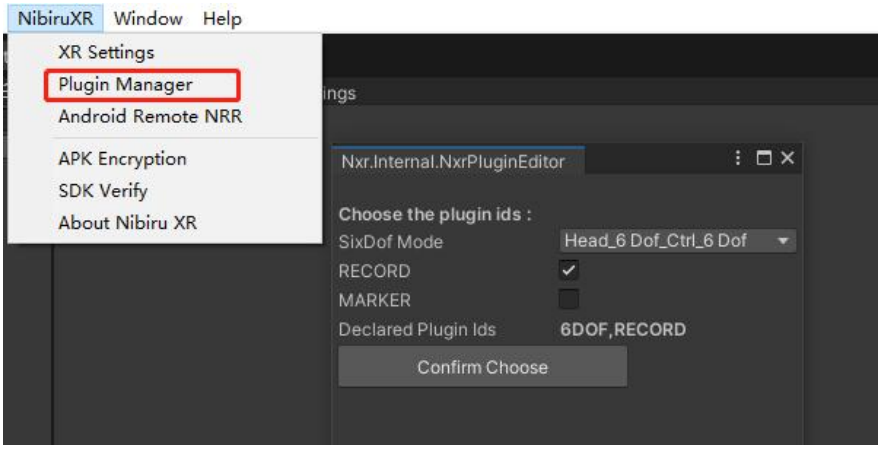


Figure 5-4 Plugin Manager Setting

5) Click “NibiruXR => SDK Verify” to set key. Fill in the key which generated by the verification tool.

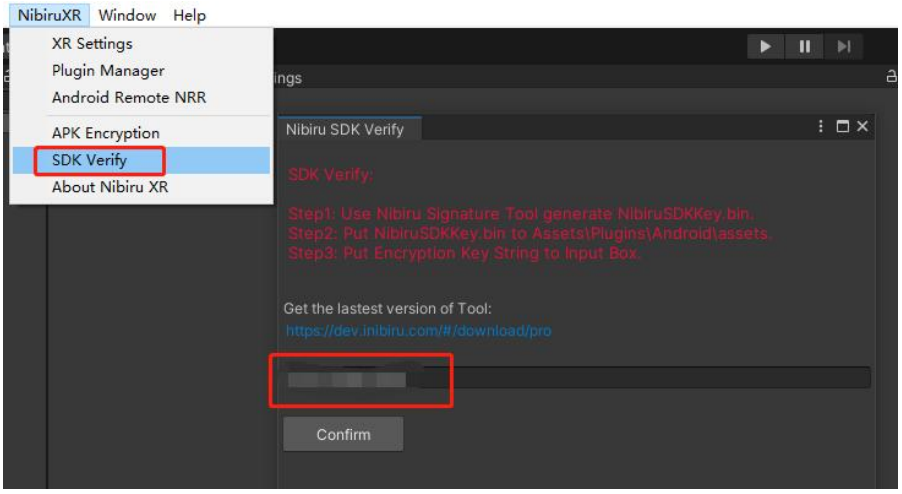


Figure 5-5 Set Key

The package name should consistent with the "Package Name" in the "Other Settings".

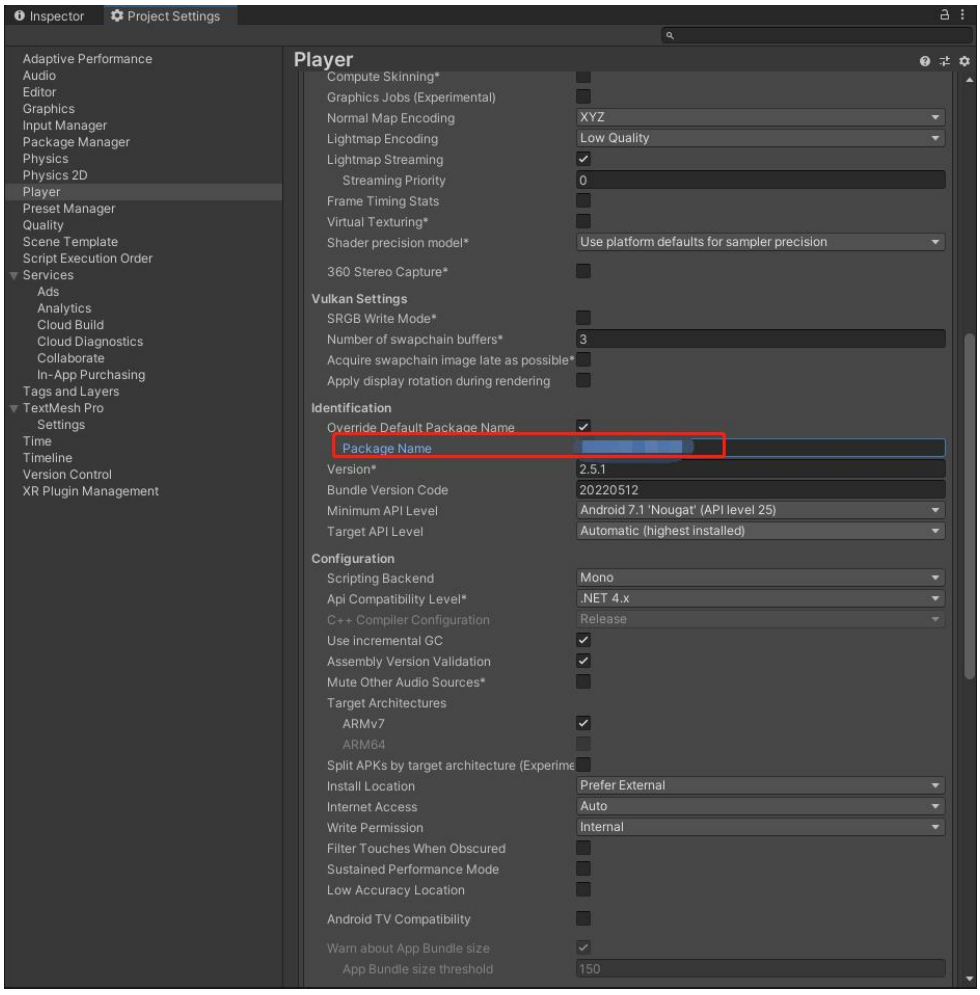


Figure 5-6 Set Package Name

6) Select platform and build APK.

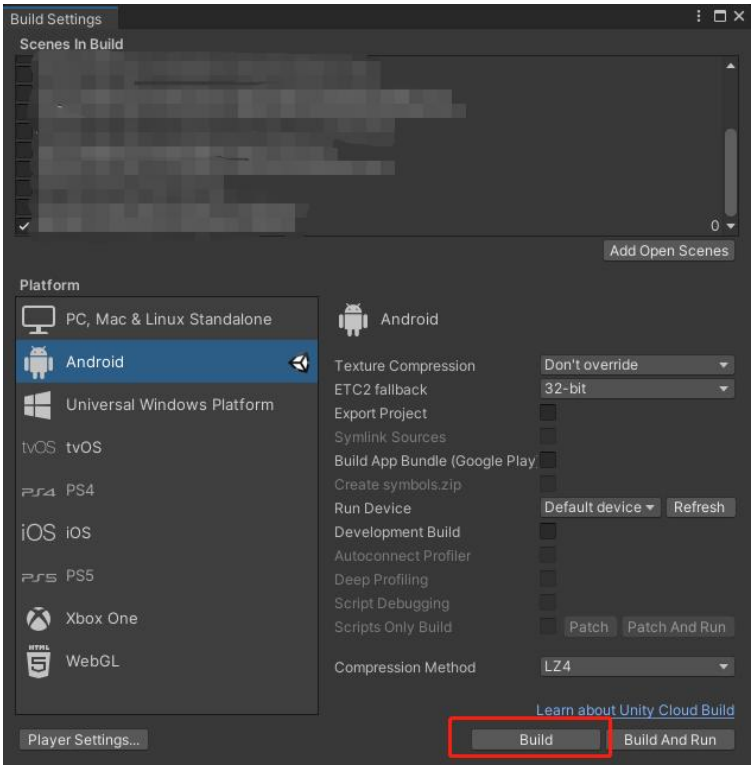


Figure 5-7 Build APK

6. Standard Specification of Content Development

1) Checking “RaycastTarget” in the image component of UGUI will consume some performance. Do not check it for saving performance. Not only the image component but also the text component has such problems. Generally, only buttons need to receive response events, so there is no need to open “RaycastTarget” for most images and text.
2) Try to reduce using “Update LateUpdate FixedUpdate” which also improves performance and saves power. Use more events (instead of SendMessage, use the event delegate written by yourself or in C#)
3) If it is not necessary, try to avoid using real-time shadows and global illumination to improve operation efficiency
4) Model triangular patch shall be controlled within 50000 monocular and the vertex number shall be controlled within 50000.
5) Try to reduce or do not using Unity light, instead of using Lightingmap information.

6) Try to reduce or do not using particle systems.

7. FAQ

1) Q: How to split the screen if the current scene contains 2D UI and an orthographic camera?

A: It is recommend to place the 2D UI to 3D scene. Secondly, it is best to keep only one main camera in the whole scene.

2) Q: Why is the split screen displayed in the editor but not in the all-in-one machine?

A: Check whether "AndroidManifest" is tagged with NVR.

Check whether the script mounted camera is "MainCamera".

3) Q: Why failed during build?

A: Check whether the jar package referenced by Android conflicts". View the Console error log in the editor, and analyze the reason.

4) Q: Why build success but run crash or black screen?

A: Open the Logcat of "Eclipse". Filter out the Unity related Log through "Tag=Unity" to check whether error occur.

If error I/Unity(4350): NullReferenceException at UnityEngine.Material..ctor (UnityEngine.Shader shader) [0x00000] appears, open "Edit->Project Settings->Graphics" and put "UnlitTexture.shader" & "SolidColor.shader" into list "Always Included Shaders" in "NAR/Resources".

Check whether the "jars" in "Plugins/Android" are duplicated.

Check the option of "PlayerSetting" should be "Multithreaded Rendering". Disable it and try again.

5) Q: Why the picture is stuck but the frame rate display is normal?

A: Check configuration of "Unity Player Settings": "Blit Type"-> "never", uncheck "Multithreaded Rendering".