stARkit SDK 用户手册

1 简介

1.1 综述

本文介绍了联想自研 AR 系统的用户层 SDK —— stARkit_device 的配置及使用方法。本文的目的是帮助使用联想自研的 AR SDK 基于联想自研的 AR 设备(dyastAR)开发 AR 应用的用户可以快速的上手，并且能够开发出让用户满意的应用。

联想 AR 框架是一种高性能; Android 上的低延迟 AR 需要在 daystAR glass 设备上访问 AR 全功能功能和软件优化。
AR 框架 SDK 提供对这些 AR 功能的访问，否则开发人员将无法使用这些功能。
此外，AR 框架 SDK 在 daystar AR glass 上实现核心低延迟渲染工作，这将帮助开发人员创建和显示他们想要的高质量 AR 内容。 哪个功能名为“显示引擎”。
核心 AR 框架 SDK 功能包括：
1: DisplayEngine
2： 3Dof 头部跟踪
3： 6Dof 位置跟踪
4： Gesture
5： 相机图像服务

2.2 SDK 包内容

2.2.1 LAR 显示引擎

全部在“LARsuite”SDK 中

/LARSuite	所有开发功能所需要的代码都在这文件夹
/Plugins/Android/libs	包含一些需要用到的so库文件
/Sample	一些利用该sdk开发的示例
Texture	设备运行时候所需要的相关图片资源

2.2.2 SLAM

2.2.2 SLAM

liblarclient.so	“LARsuite”包中的库，供开发人员从大满贯服务中调用函数 您可以通过调用so文件获取点云数据信息
liblarplugin.so	“LARsuite”包中的库，是sdk内部调用的库

2.2.3 手势服务

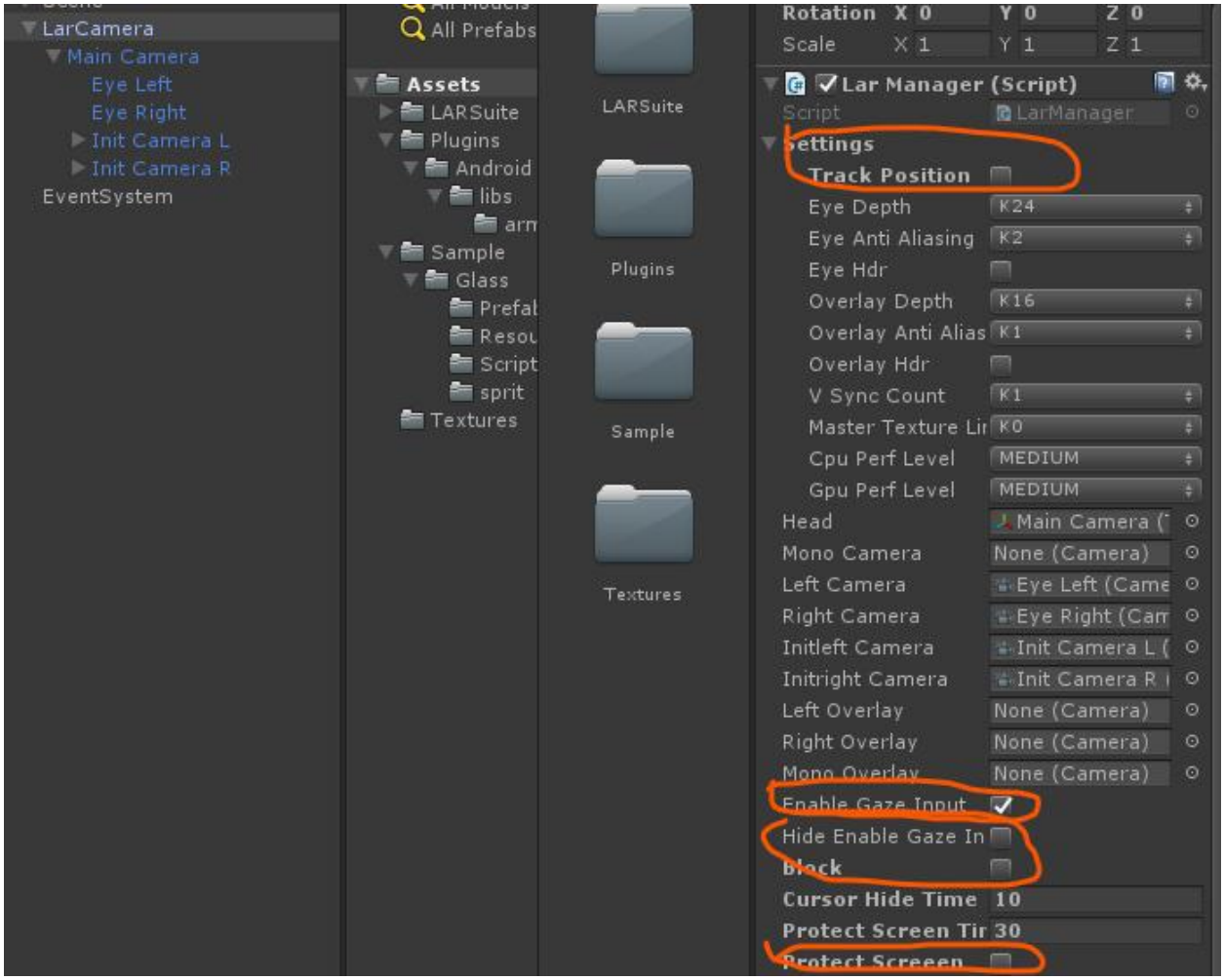
libtofclient.so	提供控制手势服务的一些方法

3.1.1 概观：

本节介绍如何将 Lenovo AR Display Engine 与我们的发行包一起使用。Lenovo Display Engine 发行包由两部分组成：Display Engine Lib 和 LarUnityPackage。Display Engine Lib 内置于设备的系统映像中。 LarUnityPackage 可以导入到 Uninty3D 并用于您的 AR 应用程序开发。3.2.2 How to use Lar Unity 包：

1. 将 stARkit unity 软件包导入 Unity3D 项目。.

2. 使用 LarCamera 而不是主摄像头。您可以在名为 LarCamera.prefab 的 Prefabs 文件夹下找到它。选择 6dof/3dof 从 LarCamera -> Settings -> Track Position
3. 选择是否让光标出现。 Enable GazeInput
4. 选择是否需要长时间不动黑屏功能 ：ProtectScreen



在 Scripts 文件夹中的三个模块：头部跟踪，输入和立体渲染。
LarCamera.prefab 上的 LarManger Script 用于根据从玻璃陀螺仪传感器读取的数据更新观察者的位置和方向。 玻璃驱动器将陀螺仪数据导出为标准的安卓陀螺数据，因此我们可以直接读取陀螺仪数据，还可以读取陀螺仪数据并更新观察者的位置和方向。

除触摸屏外，玻璃外部输入很少，Gaze 是设备中最重要的输入。您可以在 PC 中像鼠标一样使用 Gaze，只需以不同的方式移动注视光标：随头部移动。我们提供凝视输入模块来生成凝视事件并泵入统一引擎。我们现在提供四个凝视事件：凝视进入/移动/点击/存在，更多未来的凝视事件，如 Gaze Drag。我们还提供 Gaze 事件触发器来处理生成的 Gaze 事件，开发人员可以将注视事件触发器附加到 UI 控件元素并响应注视事件。 Gaze 输入模块更像普通统一应用中的通用指针输入模块。因此，现有的光标事件处理程序可以轻松地移植到凝视事件。在统一编辑器中，开发人员可以通过启用或禁用注视输入模块。
与凝视输入模块一起，在眼镜前面有一个凝视光标。注视光标有三种状态：Gaze Enter / MoveOver / Exist。当凝视光标进入或存在凝视目标时存在视觉反馈。我们开发了一个默认的 Gaze Cursor 渲染器，开发人员也可以实现自己的 GazeCursorBase 派生的凝视光标渲染。

注意事项：

我们需要按照以下设置:

1： Anisotropic Textures -> per texture
（Edit ->project settings -> quality ->rendering）

2：AA -> disabled
（Edit ->projectsettings -> quality ->rendering）

3：V Sync Count = Don'tSync （Edit ->project settings -> quality ->other）

4：Player settings –> Default orientation : landscape left

5：Player settings –> Rendering -> Multithreaded Rendering : disable

6：Player settings –> Configuration -> Write Permission : External(SDCard)

按照以下说明修改 PlaySetting 中的参数值：

应用示例：

1. 创建一个从 GazeEventTrigger 类派生的新脚本，并附加到 Cube，名为 CubeEventHandler，以使 Cube 对象接收 Gaze 输入事件。

```
public class CubeEventHandler : GazeEventTrigger {

    private Color _color;
    private MeshRenderer _render;

    void Start() {
        _render = gameObject.GetComponent<MeshRenderer>();
        _color = _render.material.color;
    }

    public override void OnGazeClick(PointerEventData data) {
        _render.material.color = Color.cyan;
    }

    public override void OnGazeEnter(PointerEventData data) {
        _render.material.color = Color.red;
    }

    public override void OnGazeExist(PointerEventData data) {
        _render.material.color = _color;
    }

    public override void OnGazeMove(PointerEventData data) {

    }
}
```

您可以收到几个 Gaze 输入事件，包括 GazeEnter，GazeClick，GazeMove 和 GazeExist。 本教程只是在 Gaze Cursor 移动并发生 Click 事件时更改立方体颜色。

您还可以使用标准的 unity 事件触发器让 gameobject 与 Gaze Input Events 交互。我们将凝视事件映射到 Unity 指针事件：指针进入凝视输入，指针点击凝视点击，指针移动到凝视移动和指针存在凝视存在。我们创建 CylinderEventHandler 到 Cylinder 来响应事件触发器。

```
public class CylinderEventHandler : MonoBehaviour {

    private Color _color;
    private MeshRenderer _render;

    void Start() {
        _render = gameObject.GetComponent<MeshRenderer>();
        _color = _render.material.color;
    }

    public void OnPointerClick() {
        _render.material.color = Color.cyan;
    }

    public void OnPointerEnter() {
        _render.material.color = Color.red;
    }

    public void OnPointerExist() {
        _render.material.color = _color;
    }
}
```

我们建议直接使用 GazeEventTrigger，它对以后的 SDK 更新很简单友好。

1. 在 Build Settings 中切换到 Android。 输入您的公司名称和产品名称，更改捆绑标识符

2. 连接设备运行

把联想设备打开，你可以看到我们创作的简单场景和凝视光标。 向左和向右移动头部，凝视光标移动，当它悬停在立方体或圆柱体上时，凝视事件触发器和游戏对象下方的颜色变为红色。 当 Gaze Click 事件发生时，它会变色。

Gaze to Canvas

场景位置: /LARSuite /Sample /Glass/ GazeToCanvas.unity

1. 创建新的 Unity 项目“GazeToCanvas”
2. 将 LARSuite SDK 导入为自定义包
3. 在空的游戏对象“场景”下创建和添加简单的 3D 对象，如立方体和圆柱体

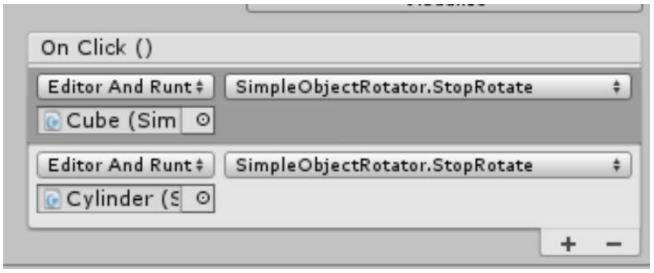
4.在其下创建 UI 画布和面板，在面板内创建一个按钮

5.找到/ Prefabs 文件夹下的预制“Lotus Application.prefab”。 拖动并将其添加到 UnityScene Hierarchy 面板。 将它放在顶级场景层次结构下。

6.找到/ LARSuite / Prefabs 文件夹下的预制件“LarCamera.prefab”。 拖动并将其添加到 Unity Scene Hierarchy 面板。 将它放在顶级场景层次结构下。

7.启用“启用注视输入”复选框

8.在 Button Inspector 中，将 Cube 和 Cylinder 添加为 Button 的 On Click Responder，



以调用停止简单对象旋转。

9：在 Build Settings 中切换到 Android。 输入您的公司名称和产品名称，更改捆绑标识符。

10：连接 Lenovo 设备和 Build＆Run。

找平面：

创建新的 Unity 项目“新项目”

1.找到 LARSuite / Prefabs 文件夹下的预制件“LarCamera.prefab”。 拖动并将其添加到 Unity Scene Hierarchy 面板。 使用 LarCamera 代替主摄像头。

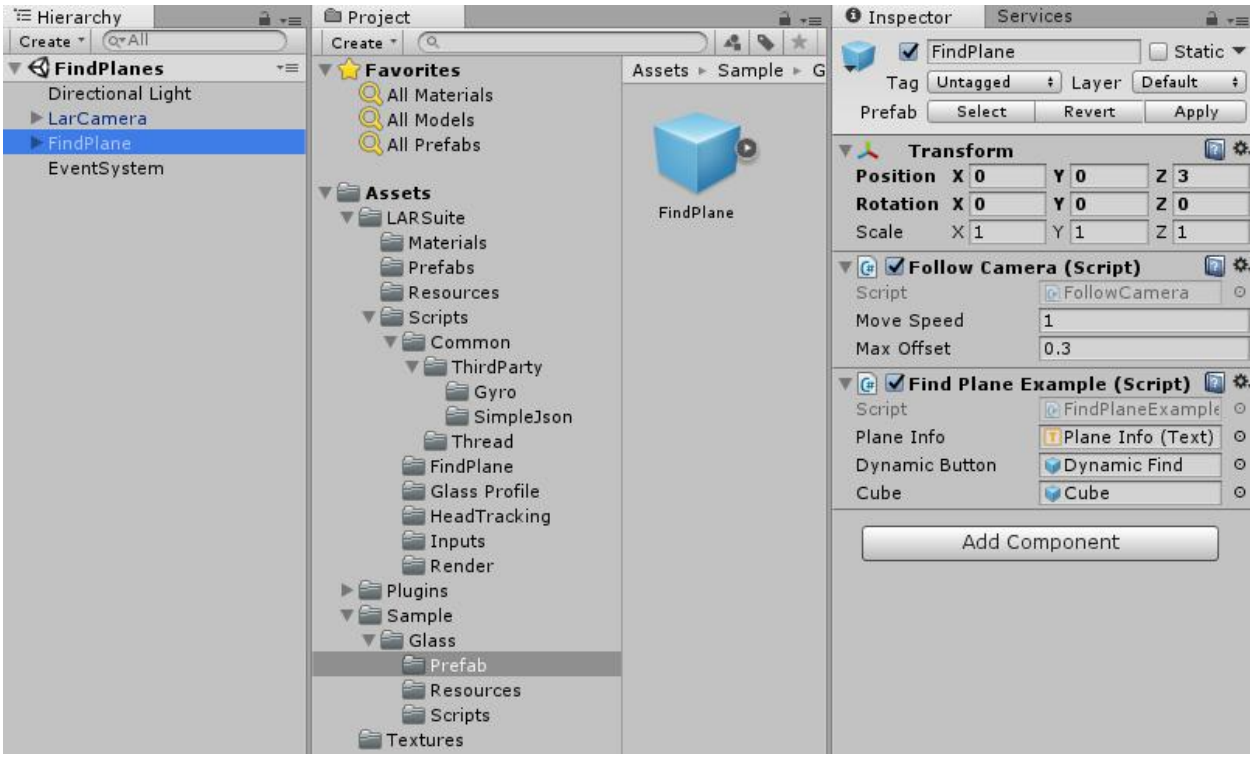
2.从 LarCamera 中选择 6dof - >设置 - >跟踪位置

3.选择 GazeInput 交互功能。

4.找到 Sample / Prefab 文件夹下的预制件“FindPlane.prefab”。 拖动并将其添加到 Unity Scene Hierarchy 面板。

6.在 Build Settings 中切换到 Android。 输入您的公司名称和产品名称，更改捆绑标识符。

7.连接 Lenovo Glass 和 Build＆Run



FindPlane 预制件上有两个脚本组件。 Follow Script 可以使 UI 界面跟随头部运动。FindPlaneExample 脚本可以实现空间平面的搜索和显示。

在 FindPlaneExample 脚本中。

1.在 Start 函数中调用“FindPlane.SetMovEnable（true）”方法

打开 TOF 相机。

2.在调用“FindPlane.StartFind（）”方法之前调用“FindPlane.SetCallBack（s_CallBack）”方法。“FindPlane.SetCallBack（s_CallBack）”可以设置一些与飞机相关的信息。

3.您可以通过“FindPlane.StartFind（）”方法开始搜索飞机

4.最后。当你想要结束对飞机的搜索时。你可以调用“FindPlane.StopFind（）;”。

Glass Profile

保存设备属性，镜头分离，fov，投影修复等.SDK 中有一个默认的配置文件。

1.创建新的 Unity 项目“GlassprofileTest”

2.将 LARSuite SDK 导入为自定义包

3.创建并添加简单的 3D 对象，如空的游戏对象“场景”下的立方体和圆柱体，您可以将简单的动画脚本附加到立方体和圆柱体。

4.找到 LARSuite / Prefabs 文件夹下的预制件“LarCamera.prefab”。拖动并将其添加到 Unity Scene Hierarchy 面板。 将它放在顶级场景层次结构下。

5.创建一个新的脚本类并附加到 LarCamera，命名为 readGlassProfile

您可以通过此方法获取自定义配置文件：“GlassProfileManager.Instance.CustomProfile（）”

您可以通过以下方法获取默认配置文件：“GlassProfileManager.Instance.DefaultProfile（）”

创建一个类来接收它：

GlassProfile glassProfile = GlassProfileManager.Instance.CustomProfile（）；

然后你可以在配置文件中获取参数：

```
GlassProfile glassProfile= GlassProfileManager.Instance.CustomProfile ();
Debug.Log("LenseSeperationLeft" + glassProfile.LenseSeperationLeft);
Debug.Log("LenseSeperationLeft" + glassProfile.LeftProperty.PosX
    + " " + glassProfile.LeftProperty.PosY + " " + glassProfile.LeftProperty.PosZ);
Debug.Log(glassProfile.LeftProperty.RotX + " " + glassProfile.LeftProperty.RotY
    + " " + glassProfile.LeftProperty.RotZ);

Debug.Log(glassProfile.LenseSeperationRight);
Debug.Log("LenseSeperationRight" + glassProfile.RightProperty.PosX
    + " " + glassProfile.RightProperty.PosY + " " + glassProfile.RightProperty.PosZ);
Debug.Log("LenseSeperationRight" + glassProfile.RightProperty.RotX
    + " " + glassProfile.RightProperty.RotY + " " + glassProfile.RightProperty.RotZ);
Debug.Log(glassProfile.LenseFrustumData.Fov);
```

您可以通过以下方式获取屏幕的宽度和高度：

Vector2 ScreenSize = glassProfile.ScreenSize;

你可以通过这个获得相机的 FOV：

Int fov = glassProfile.LenseFrustumData.Fov

最后，您可以使用获取的数据设置摄像机参数。

PointCloud

你可以获取相关的点云数据通过调用 so 库文件

```
[DllImport("larclient")]
public static extern bool start_slam();
[DllImport("larclient")]
public static extern bool stop_slam();
[DllImport("larclient")]
public static extern int slam_init();

[DllImport("larclient")]
public static extern bool mode_switch(bool mode);
[DllImport("larclient")]
public static extern void get_point_cloud(out int len, IntPtr data);

GCHandle dataObject;
IntPtr dataInptr;
```

手势：

您可以获取有关当前设备和手势功能相关的信息：

```
ture;
[DllImport("tofclient")]
public static extern void movEnableDisable(bool enable);
[DllImport("tofclient")]
public static extern int getMovStatus();
[DllImport("tofclient")]
public static extern void  tofStatusCallback(TofStatusCallback status);


public delegate void TofStatusCallback(int tLen);
public static TofStatusCallback statusCallback;
```

获取当前 sdk 版本号：

可以通过 GetVersion 类中的 GetVersionfun()方法获取当先的版本号：

```
public class GetVersion  {


    public static string GetVersionfun( )
    {
        int major = LarManager.starkitVersion_major;
        int minor = LarManager.starkitVersion_minor;
        string versionString = "version is :" + major + "." + minor;
        return versionString;
    }

}
```