
ZED
unity

ZED Wrapper for Unity by Stereolabs

Introduction

This package is an interface to access most functions of the ZED SDK in Unity. It includes ZED stereo images, depth map and positional tracking.

Installation

This section details the installation process of the Unity package inside a scene. If you used a previous package, please remove it from your assets before installing this one.

Content

This package contains 4 examples scenes in ZED/Scenes:

- SimpleMR, shows how to integrate virtual objects the real world.
- SimpleTracking, extracts position and orientation of the ZED camera in space.
- GreenScreen, shows how to use the green screen prefab.
- TrackingVR, is a little game using the HMD and the ZED's tracking. The virtual reality mode needs to be checked before using this scene.

List of prefabs available :

- A prefab for tracking is available in the ZED/prefabs folder called ZED_Tracking.
- A prefab for basic mixed reality functionalities called ZED_Rig.
- A prefab for a basic green screen called ZED_GreenScreen.

Requirements

- Requires Unity 5.5.X or higher. Prefabs might be broken with earlier versions.
- ZED SDK 2.0
- DirectX 11

If you have another version of the ZED SDK, this plugin may not work.

Create your project in Unity

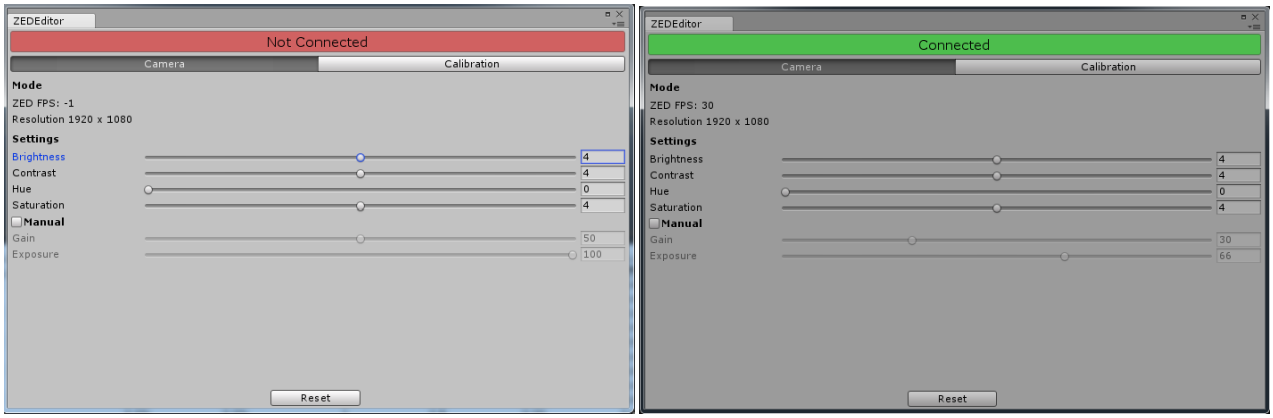
- When Unity is started, create a new project, enter a name and click on "Create Project"
- In the menu bar, click on Assets >> Import Package >> Custom Package… and select the ZED Unity package file (*.unitypackage). The unity import window should appear and click on "import".
- In the project window, select one of the provided scene located in ZED/Scenes/SimpleMR (Images, depth and tracking) or ZED/Scenes/SimpleTracking (Tracking only) by double clicking on it. You can also use the prefabs
- A simple scene with spheres and cubes should appear in the Scene and Game window.
- Click on play button to launch it.

ZED Settings in Unity

A tool called ZED Camera is available in Unity. Go to Window ZED Camera. It displays camera calibration parameters and allows you to adjust ZED camera settings.

Camera settings

The ZED settings panel allows you to adjust different variables such as Brightness, Contrast, Hue, Saturation, Gain, Exposure and white balance. The FPS displays ZED video framerate and will not affect Unity rendering FPS.



The camera calibration parameters (focal length, optical center, …) are also displayed in the Calibration tab as they would be accessed through the C++ ZED SDK getParameters() function.

Interfacing the ZED with Unity

A ZEDManager.cs file is available to interface the ZED with Unity. It allows you to access ZED images, depth and tracking. A SVOManager.cs file is also available, it enables to read and record video. But if you need your own manager or integrate it inside another script, the next section explains how to do it.

ZED's init

Firstly you need to initialize the camera.

```
// Retrieves an instance of the camera, only one ZEDCamera can exist.
ZEDCamera zed = ZEDCamera.GetInstance ();

// Creates a camera if it has not been made yet, here we create a camera with a HD1080 resolution
//It's possible to get other resolutions, such as VGA, HD720 and 2k.
zed. CreateCamera ( ZEDCamera.ZEDResolution_mode.HD1080)

// Initializes the camera in performance mode, and have meters as unit.
// The ERRCODE is equal to SUCCESS if the camera has been well initialized.
ZEDCamera.ERRCODE e = zed.Init ( ZEDCamera.MODE.PERFORMANCE );
```

The following sections will explain how to use the tracking, and how to use the images from the ZED.

Tracking

Positional tracking allows you to get camera position and orientation in space. The tracking is on the left camera.

```
// Pos is an array of float, or a matrix which contains the first position
bool tracking = zed.EnableTracking (ref pos, ref rotation );
if (! tracking ) throw new Exception ("Error , tracking not available ");
```

To extract ZED position, you have to call `zed.Grab()` then:

```
//This structure will contains all the data concerning the current position
sl.zed.MotionPoseData motion = new sl.zed.MotionPoseData();
//...
// Get the position of the ZED, it fills the structure "motion" with the translation and rotation and c
zed. GetPositionCamera (ref motion);
// Set the new position in local, to not erase user hand made position
transform.localRotation = motion.rotation;
transform.localPosition = motion.translation;
```

To move the ZED in a mixed reality environment, you need to change the projection matrix of the current camera to the ZED matrix.

```
// mainCamera is the object Camera of Unity
mainCamera.projectionMatrix = ZEDCamera.GetInstance ().Projection();
```

And change the field of view of the current camera.

```
//Set the new FOV
mainCamera.fieldofview = ZEDCamera.GetInstance().GetFOV()*Mathf.Rad2Deg;
```

More information is available here.

Retrieve Images

Now let's retrieve ZED images and convert them to textures.

```
//Calls Grab, all the computation are made by this function
if ( zed.Grab ( ZEDCamera.SENSING_MODE.FILL ) == 0) {
    // Fills the different textures if the grab has succeeded
    zed.UpdateTextures ();
}
```

Now all the textures are computed, you can retrieve them :

```
// Gets the images from the left camera , this texture will be updated as long as zed.image.UpdateTextu
// No need to create multiple instances of the same texture, only one instance of texture type can be c
Texture2D leftSide = zed.CreateTextureImageType ( ZEDCamera.SIDE.LEFT );
```

Each texture created is on GPU and it enables a fast computation on shader or directCompute.

You can extract different image formats from the ZED:

- CreateTextureImageType corresponds to retrieveImage from C++
- CreateTextureViewType corresponds to getView from C++
- CreateTextureMeasureType corresponds to retrieveMeasure from C++

More information about each type of image is available here.

Depth Map

The ZED outputs a real-time depth map of the scene.
You can access it in Unity with the `computeDepthXYZ` function available inside S_ZED_Utils.cginc.

To access the ZED shaders, on the inspector window, right click on the Mat_ZED_RGB_Depth shader and select "Edit Shader".

GreenScreen

Set the scene

To use the GreenScreen feature, drag and drop a prefab located at Assets/ZED/Prefabs called ZED_GreenScreen in your scene.

To make the camera move with a vive pad you need to add a new GameObject as a father, add a component called Steam VR_Tracked Object and finally set the right device.

The object virtual and real may not match directly, you need to put the pad tracked at the right position from the ZED.

Make adjustements

To get a full screen go to the component Screen Manager RGB Depth located in the GameObject Displayer then hit the button "Open camera preview" to open a preview. A checkbox full screen appears below, check it to get a full screen. To exit the full screen press "escape".

If the colors in the preview are too bright or dark check/uncheck the box "Color booster".

The GreenScreenManager is a script which enables to change the settings to control the color which will be transparent.

General options

The script enables to show different views:

- FINAL : The final effect
- FOREGROUND : The output of the camera
- BACKGROUND : The virtual
- ALPHA : The transparency in a RGB format (greyscale)
- KEY : The transparency multiplied by the Source

ChromaKey

A chroma key is the color which will be removed. To get the color, use the color picker then set the Range (the closeness between two colors) and the Smoothness with the sliders. Unity per default reset all values made during the play of the game, but it's possible to save your last changes by trigerring the button "save", and reload them with the button "load".

If you only get the exe and if the developper had used the prefab ZED_GreenScreen in it's scene, you have access to the greenScreen options by creating a file "shader_greenScreen_save" and put it next to the exe. The file should have this form and its called shader_greenScreen_save per default :

```
{"chromaKeys":{"color":{"r":0.0,"g":1.0,"b":0.0,"a":1.0},
"smoothness":0.07999999821186066,
"range":0.4000000059604645},
"numberColors":1,"erosion":0,
"numberBlurIterations":5,"blurPower":0.10000000149011612,
"whiteClip":1.0,
"blackClip":0.0,
"spill":0.10000000149011612}
```