Copernic360 User Guide

http://copernic360.ai

KageNova
http://www.kagenova.com

January 21, 2020

Contents

1 Overview

Copernic360 enables 6 degree-of-freedom (6DOF) motion in standard 360° VR content to allow users to move freely in scenes. It consists of two components: a Unity plugin; and a Cloud API.

The Unity plugin enables 6DOF on your scene's skybox. In order to do so, the plugin requires a configuration file with metadata for your content. This configuration file is computed by preprocessing your content through our Cloud API.

The plugin is designed to be very easy to integrate into existing 360° content viewers, particularly if you are currently rendering the content on a skybox.

This document describes how to integrate the copernic360 Unity plugin into your Unity project.

Integrating the plugin requires two or three steps, depending on whether you want to display static images only or videos as well.

2 Skybox

Replace your scene's skybox (via Lighting settings) with the one provided at `copernic360/Materials/6DoFSkybox`. There's no need to set the material's texture to your 360° content, this will be done at runtime instead.

3 Instantiating a skybox controller

When your scene loads (either in an `Awake` method or in response to a `sceneLoaded` event), you need to instantiate a `SkyboxController` (found in the `Kage.Copernic360` namespace). This class is *not* a `MonoBehaviour` and consequently does not live in the scene graph.

The `SkyboxController`'s constructor takes two arguments:

1. The 360° texture to be rendered. For images this would normally just be the texture of the image itself. For videos, you should set up your video player to render to a `RenderTexture` and pass that render texture to the skybox controller.
2. The configuration data for this piece of content. This is either an instance of `ImageConfiguration` or `VideoConfiguration` depending on the type of content (these classes are found in the `Configuration` subnamespace). Both of these classes come with static helper methods called `LoadFromStream` that let you create an instance given a stream, e.g. a `FileStream`. See the Cloud API section of the docs for how to obtain the actual files.

For image content, this is all you need to do. Here is an example of what your component could look like:

```
using Kage.Copernic360;
using Kage.Copernic360.Configuration;
using System.IO;
using UnityEngine;

public class Copernic360ImageLoader : MonoBehaviour
{
    [SerializeField] private Texture2D skyboxTexture
        = Texture2D.whiteTexture;

    private SkyboxController skyboxController;

    private void Awake()
    {
        string configPath = Path.Combine(
            Application.streamingAssetsPath,
            "copernic360/example-config.6dof");

        using (var fs = new FileStream(configPath,
            FileMode.Open, FileAccess.Read))
        {
```

```
            skyboxController = new SkyboxController(
                skyboxTexture,
                ImageConfiguration.LoadFromStream(fs)
            );
        }
    }
}
```

Note that the above script doesn't work on Android, since streaming assets cannot be accessed as simple files there. Instead, you would need to read the file via `UnityWebRequest` and wrap the resulting data in a `MemoryStream`. The plugin comes with an example script which shows how to do this.

4 Feeding frame data to copernic360 (videos only)

For video content, the skybox controller needs to know which frame is currently being displayed, since the configuration can change from frame to frame. This is done through the `SetCurrentFrame` method. You can either call this method every frame from an `Update`, or you can register to your video player's equivalent of `frameReady` to save on unnecessary calls if the video is paused or runs at a slower frame rate than your application.

Here is an example of what your component could look like for video content, assuming you're using Unity's built-in `VideoPlayer`:

```
using Kage.Copernic360;
using Kage.Copernic360.Configuration;
using System.IO;
using UnityEngine;
using UnityEngine.Video;

public class Copernic360VideoLoader : MonoBehaviour
{
    [SerializeField] private VideoPlayer videoPlayer;

    private SkyboxController skyboxController;

    private void Awake()
    {
        string configPath = Path.Combine(
            Application.streamingAssetsPath,
            "copernic360/example-config.6dof");

        using (var fs = new FileStream(configPath,
            FileMode.Open, FileAccess.Read))
        {
            skyboxController = new SkyboxController(
                videoPlayer.targetTexture,
                VideoConfiguration.LoadFromStream(fs)
            );
        }
```

```
    }

    private void Update()
    {
        long frame = videoPlayer.frame;
        if (frame < 0)
            return;

        skyboxController.SetCurrentFrame((ulong)frame);
    }
}
```

5 Rotating the scene

By default, your 360° scene is oriented such that edge of the texture lies on the negative z-axis. You can change this with the `SetRotation` method of the skybox controller. Just pass in the required clockwise rotation in degrees. E.g. to put the texture's edge on the negative x-axis, you could use:

```
skyboxController.SetRotation(90);
```

6 Support

For technical support please email support@kagenova.com and we will do our best to help!