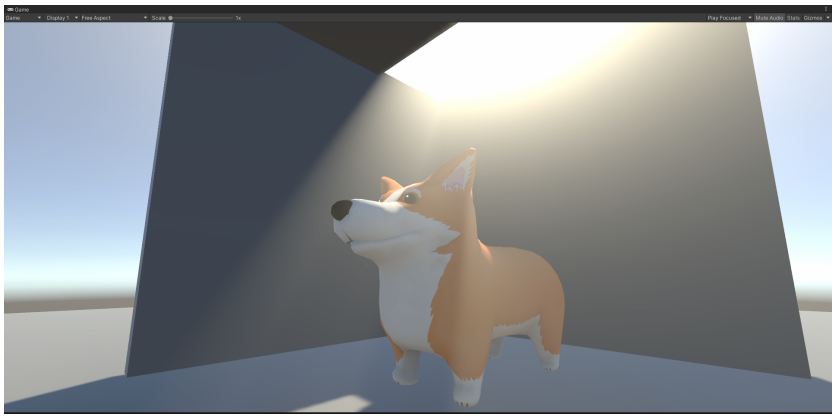
Corgi God Rays

Volumetric Lighting in URP

latest documentation here

[demo here]



Info

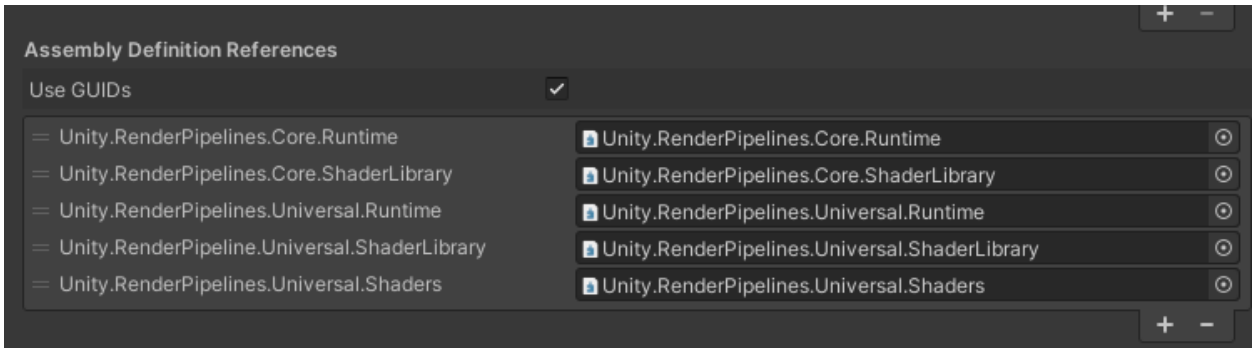
Quickly and easily drop in volumetric lighting for URP.

Features

- Very simple drop in. Add RenderFeature and add Post Process Volume Component and BAM, immediately have God Rays that you can edit in real time.
- Highly optimized volumetric lighting system, with some options for making it even faster if necessary.

Requirements

- Unity 2020.1 or higher.
- Universal Render Pipeline package: `com.unity.render-pipelines.universal`



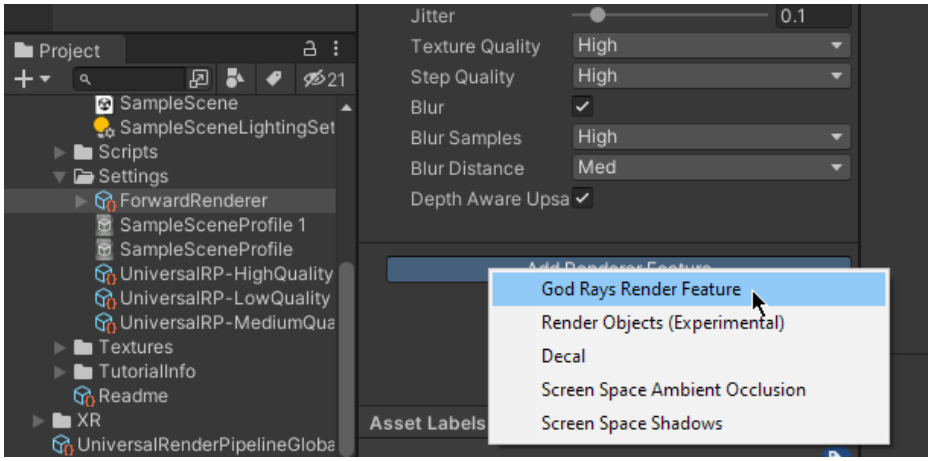
Getting Started

In order to use Corgi God Rays, you must be using Unity 2020 or above with URP installed.

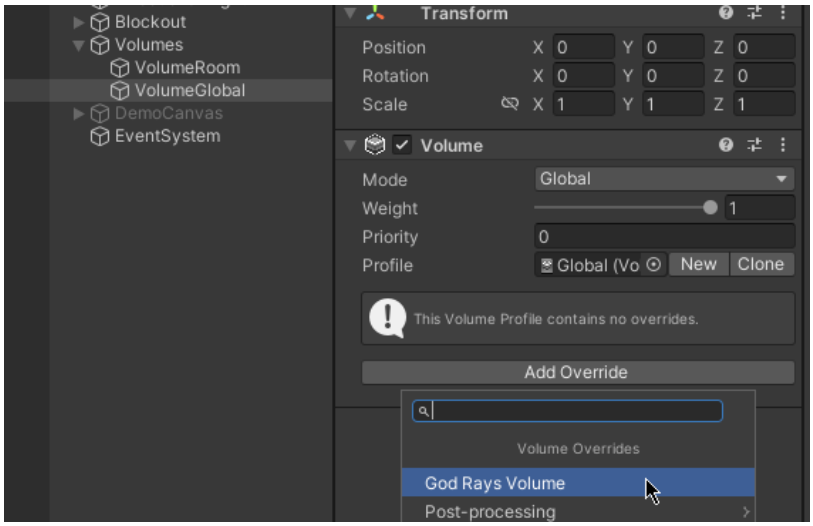
This system has two parts:

1. The Unity URP RenderFeature, to render the effect or setup material stuff.
2. The ‘post-process’ VolumeComponent.

So to get started, navigate to your project’s URP Renderer Data and add the RenderFeature.



Once this RenderFeature has been added to your renderers, you can add a VolumeComponent to any ‘post-process’ Volume in your scene. In this example, I’m adding it to a global volume.



The color of the volumetric lights should match the color of the real lights in the scene.
If nothing shows up for you, please verify that your camera and URP settings have post processing enabled.

Configuring the VolumeComponent

Here’s a screenshot of the post process volume component.

Intensity

This is the overall intensity of the effect. Useful for making areas feel more or less foggy.

Scattering

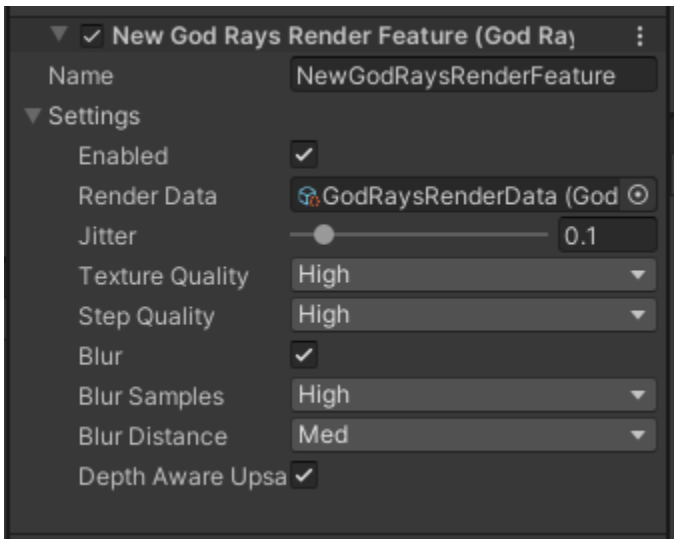
This is the scattering effect of the lights, used when creating the internal volumetrics texture. Higher values will spread out and lessen the effects of volumetric lights.

Tint

This is a multiply against the final volume color.

Configuring the RenderFeature

Here's a screenshot of the RenderFeature.



Jitter

This slider controls the internal 'jitter' used to lessen the banding visuals when using lower quality settings. If you have blur disabled, you'll notice it causes a lot of noise. The blur is intended to smoothen this out. The default value should be fine for most use cases, but the slider is here in case you need it.

Texture Quality

The quality of the internal volumetrics texture. When high, it will match the resolution of the game. When medium, it will be ½ the resolution. When low, it will be ¼ the resolution.

Step Quality

The number of steps used when tracing to create the internal volumetrics texture. Higher values will have higher quality and less banding, but of course take much more GPU time to compute.

Blur

When using lower quality settings, a blur can be useful for making the effect still look nice. Blurring internally uses a bilateral blur. It's depth-aware, so hard edges are hopefully preserved even after blurring.

Blur Samples

The number of samples taken when blurring. Higher values are more accurate, but more expensive on the GPU.

Blur Count

Higher values will lead to a more blurry image. Has no effect on GPU performance.

Depth Aware Upsampling

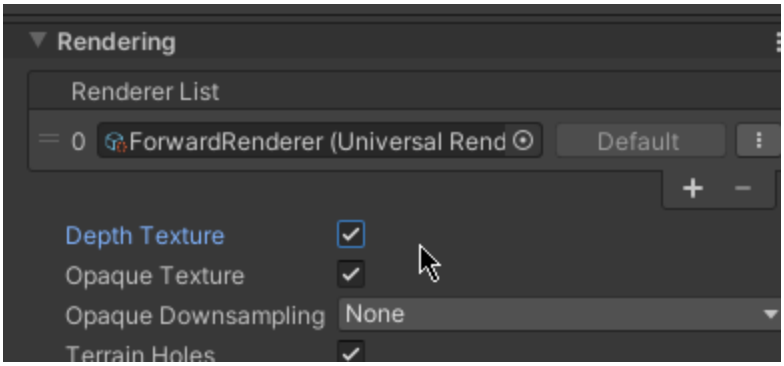
Should almost always be toggled on. Drastically improves visuals when texture quality is not at the max value.

Allow Additional Lights

Allows additional light data to be pulled from URP into the plugin for making them volumetric as well.

Use Unity Depth Directly

You may need to toggle this on or off depending on your exact SRP configuration, but usually you'll want to just leave it on. When this is enabled, you may need to enable your renderer's quality setting to have Depth Texture checked on.



Support Unity Screen Space Shadows

If you are using Unity's ScreenSpaceShadows RenderFeature, you may need to enable this. If not, you usually will want to leave this disabled. The reason this is necessary is because when the screen space shadows pass runs it sets up keywords in a way that would make tracing through the world's shadow data impossible. Fixing this incurs a slight CPU overhead because it requires doing some c# reflection to get data normally internal in Unity, because they love making things difficult for their users.

Render After Transparents

Reconfigures the god rays to calculate after transparent objects have been rendered. You may need to toggle this for compatibility with other plugins. I recommend leaving this off if possible.

Render Data

This is an internal reference data block. It should be automatically set and does not normally need to be changed. If you want to customize the asset, feel free to swap out any material references inside.

Contact Me!

If you run into any issues or just need some help, feel free to reach out and contact me.

- You can email me at coty@wanderingcorgi.com
- Or you can hit me up on Discord (Coty#2845).
- There's also a support discord here: https://discord.gg/n23MtuE