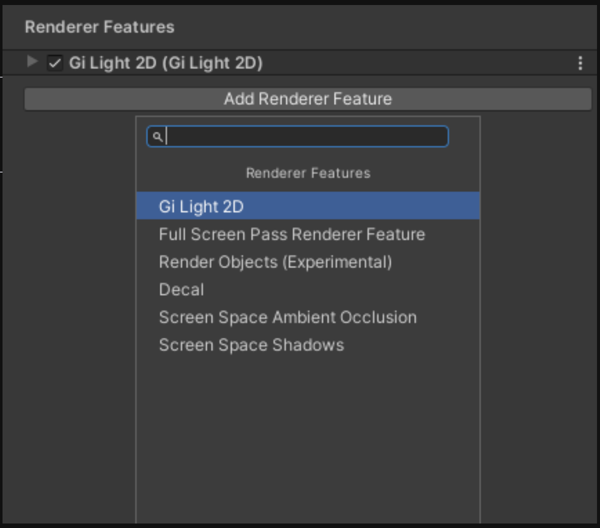
GiLight2D

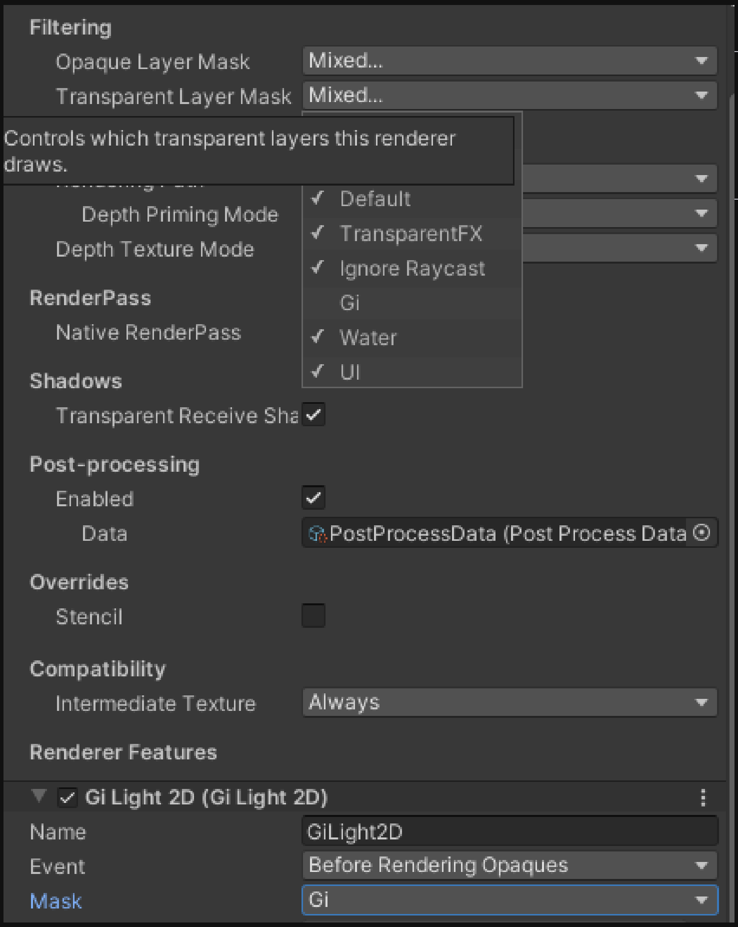
2D raytracing and artistic tool fot Unity Urp

This is a short manual consisting of a quick guide and a brief explanation of what the lighting settings are responsible for.

Quick Guide

Add GiLight2D RenderFeature to UrpRendererAsset


Renderer Features
Gi Light 2D (Gi Light 2D)
Add Renderer Feature
Renderer Features
Gi Light 2D
Full Screen Pass Renderer Feature
Render Objects (Experimental)
Decal
Screen Space Ambient Occlusion
Screen Space Shadows



Filtering
- Opaque Layer Mask: Mixed...
- Transparent Layer Mask: Mixed...

Controls which transparent layers this renderer draws.

- ✓ Default
- ✓ TransparentFX
- ✓ Ignore Raycast
- Gi
- ✓ Water
- ✓ UI

Depth Priming Mode
Depth Texture Mode

RenderPass
- Native RenderPass

Shadows
- Transparent Receive Sha ✓

Post-processing
- Enabled ✓
- Data: PostProcessData (Post Process Data

Overrides
- Stencil

Compatibility
- Intermediate Texture: Always

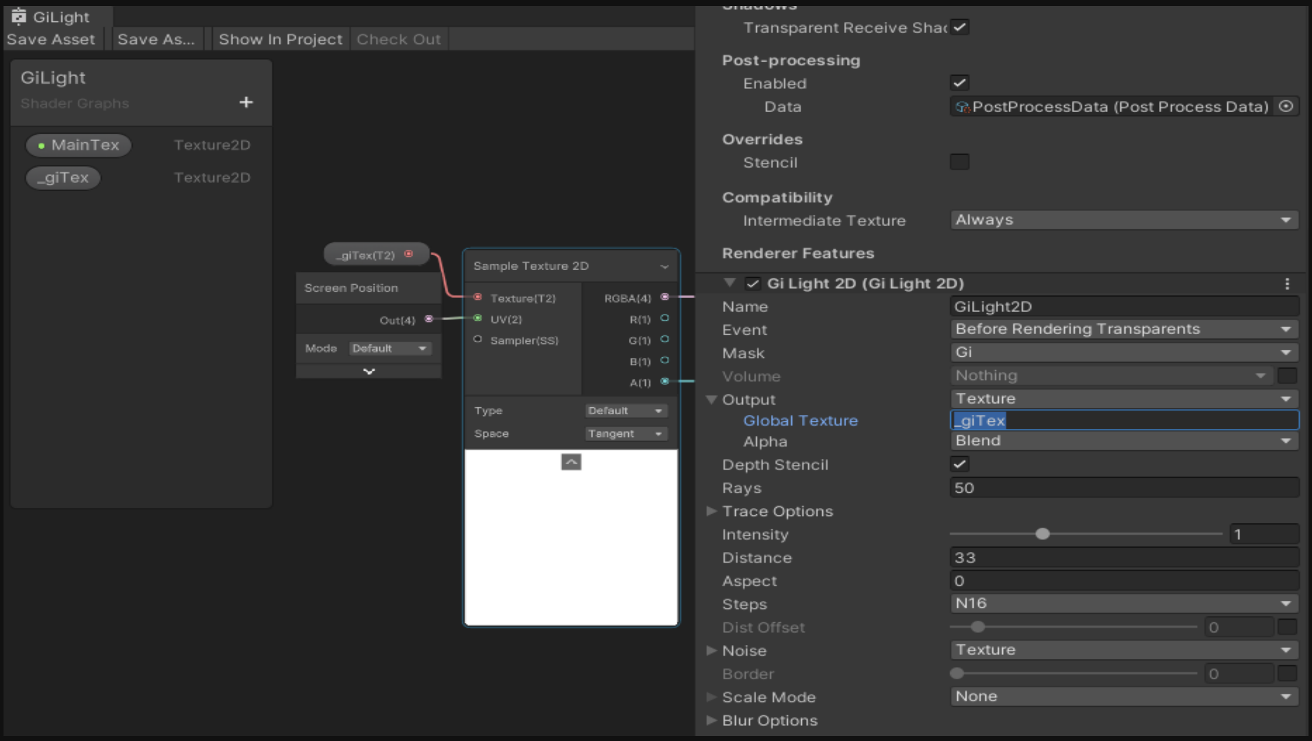
Renderer Features

✓ Gi Light 2D (Gi Light 2D)

Name	GiLight2D
Event	Before Rendering Opaques
Mask	Gi

Configure render Mask for objects you want to use for Gi Calculation

Define the output and other settings


GiLight
Save Asset | Save As... | Show In Project | Check Out

GiLight
Shader Graphs

MainTex	Texture2D
_giTex	Texture2D

_giTex(T2) → Sample Texture 2D: Texture(T2)
Screen Position: Out(4) → UV(2); Mode: Default
Sample Texture 2D: Texture(T2), UV(2), Sampler(SS); RGBA(4), R(1), G(1), B(1), A(1); Type: Default; Space: Tangent

Transparent Receive Shadows: ✓

Post-processing
Enabled: ✓
Data: PostProcessData (Post Process Data)

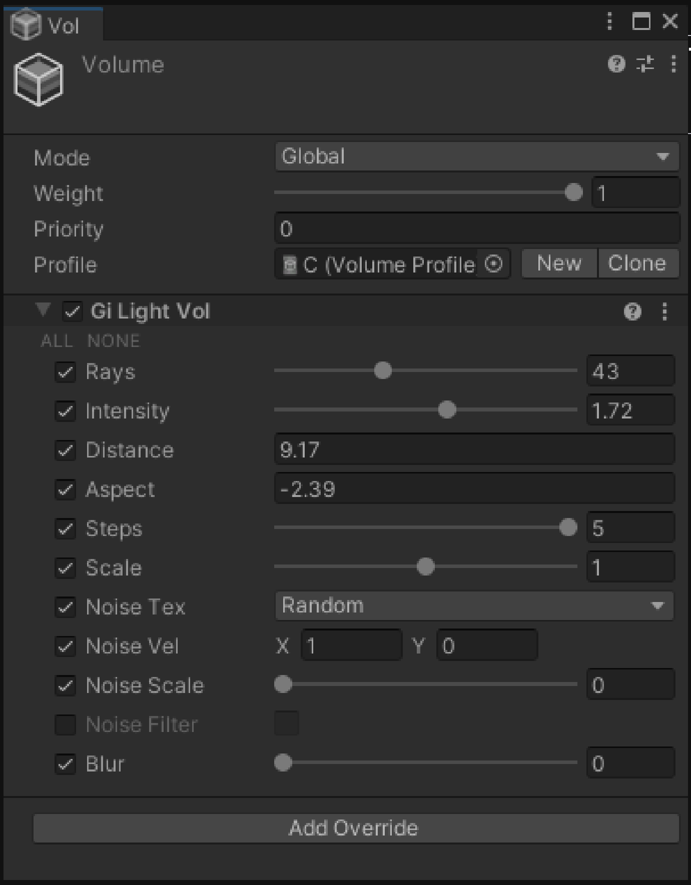
Overrides
Stencil: ☐

Compatibility
Intermediate Texture: Always

Renderer Features

Gi Light 2D (Gi Light 2D) ✓

Name	GiLight2D
Event	Before Rendering Transparents
Mask	Gi
Volume	Nothing
Output	Texture
Global Texture	_giTex
Alpha	Blend
Depth Stencil	✓
Rays	50
Trace Options	
Intensity	1
Distance	33
Aspect	0
Steps	N16
Dist Offset	0
Noise	Texture
Border	0
Scale Mode	None
Blur Options	


Vol
Volume

Mode	Global
Weight	1
Priority	0
Profile	C (Volume Profile) — New / Clone

Gi Light Vol ✓
ALL NONE

✓ Rays	43
✓ Intensity	1.72
✓ Distance	9.17
✓ Aspect	-2.39
✓ Steps	5
✓ Scale	1
✓ Noise Tex	Random
✓ Noise Vel	X 1 Y 0
✓ Noise Scale	0
☐ Noise Filter	☐
✓ Blur	0

Add Override

The settings can be controlled via PostProcess Volume

How it Works?

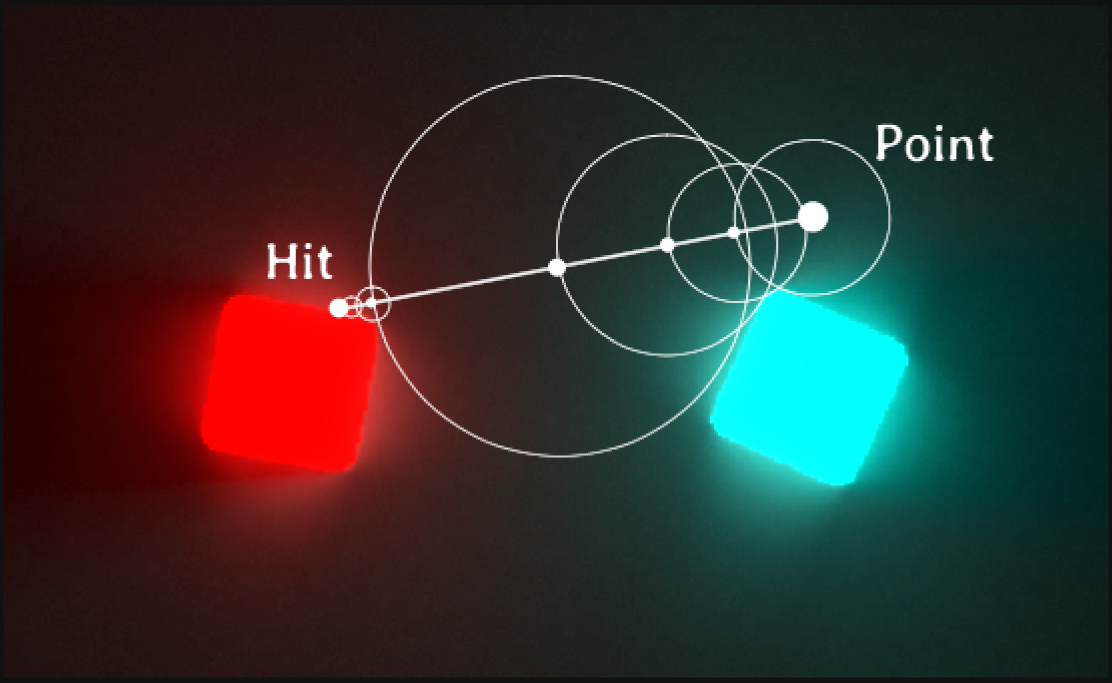
(information to understand the impact of settings)

Render basically works as post processing for filtered by layer mask objects and generates lighting texture as output.

To do this three textures are used for rendering:

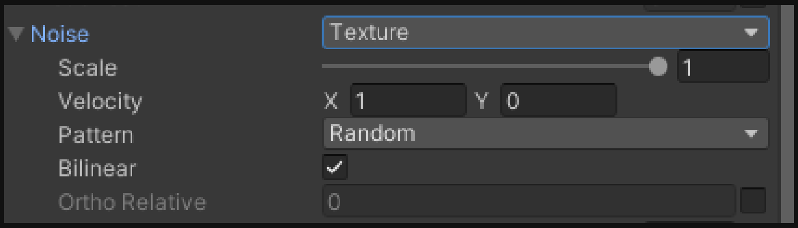
- **Object** - light sources, alpha affects to color impact
- **Noise** - a texture that shifts the initial angle of emitted rays to smooth out linearity, can create distortions
- **Distance** - a texture for optimizing ray searching, work with offset to adjust ray marching and deform lighting

Rays are emitted from each pixel in all directions, and when they hit an object, their color, dependent on distance and brightness, is added to the final pixel color.



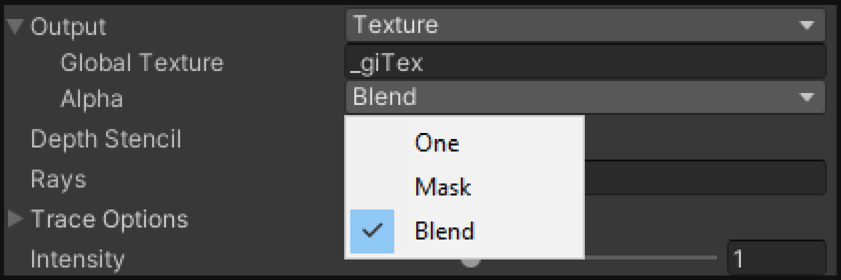
By controlling the maximum number of ray steps noise texture resolution and type, interesting effects can be achived.

Nose texture can be configured in nose setting. With the low number of rays it can create interesting effects distortion effects and abstractions.



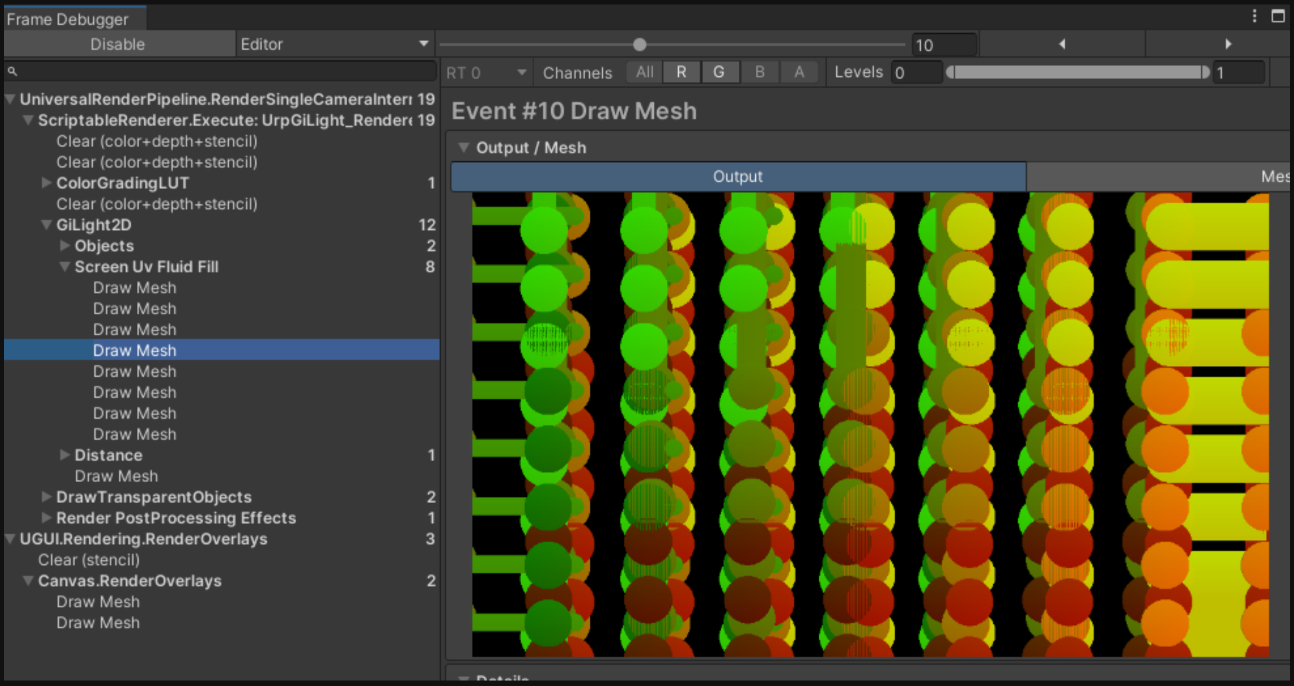
```
▼ Noise            Texture ▾
    Scale          ———————————● 1
    Velocity       X 1   Y 0
    Pattern        Random ▾
    Bilinear       ✓
    Ortho Relative 0
```

The result can be drawn directly in camera or a texture, with diferent information in alpha channel



```
▼ Output           Texture ▾
    Global Texture _giTex
    Alpha          Blend ▾
  Depth Stencil      One
  Rays               Mask
▶ Trace Options    ✓ Blend
  Intensity        ———————— 1
```

In the Unity Frame Debugger, you can take a closer look at the rendering process.

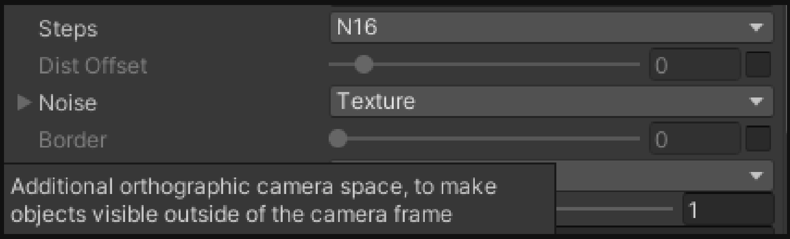


```
Frame Debugger
Disable    Editor ▾                      10
RT 0 ▾  Channels  All R G B A  Levels 0  1

▼ UniversalRenderPipeline.RenderSingleCameraInter  19
  ▼ ScriptableRenderer.Execute: UrpGiLight_Render  19
      Clear (color+depth+stencil)
      Clear (color+depth+stencil)
    ▶ ColorGradingLUT                               1
      Clear (color+depth+stencil)
    ▼ GiLight2D                                    12
      ▶ Objects                                     2
      ▼ Screen Uv Fluid Fill                        8
          Draw Mesh
          Draw Mesh
          Draw Mesh
          Draw Mesh
          Draw Mesh
          Draw Mesh
          Draw Mesh
          Draw Mesh
      ▶ Distance                                    1
        Draw Mesh
    ▶ DrawTransparentObjects                        2
    ▶ Render PostProcessing Effects                 1
▼ UGUI.Rendering.RenderOverlays                     3
    Clear (stencil)
  ▼ Canvas.RenderOverlays                           2
      Draw Mesh
      Draw Mesh

Event #10 Draw Mesh
▼ Output / Mesh
Output | Mes
```

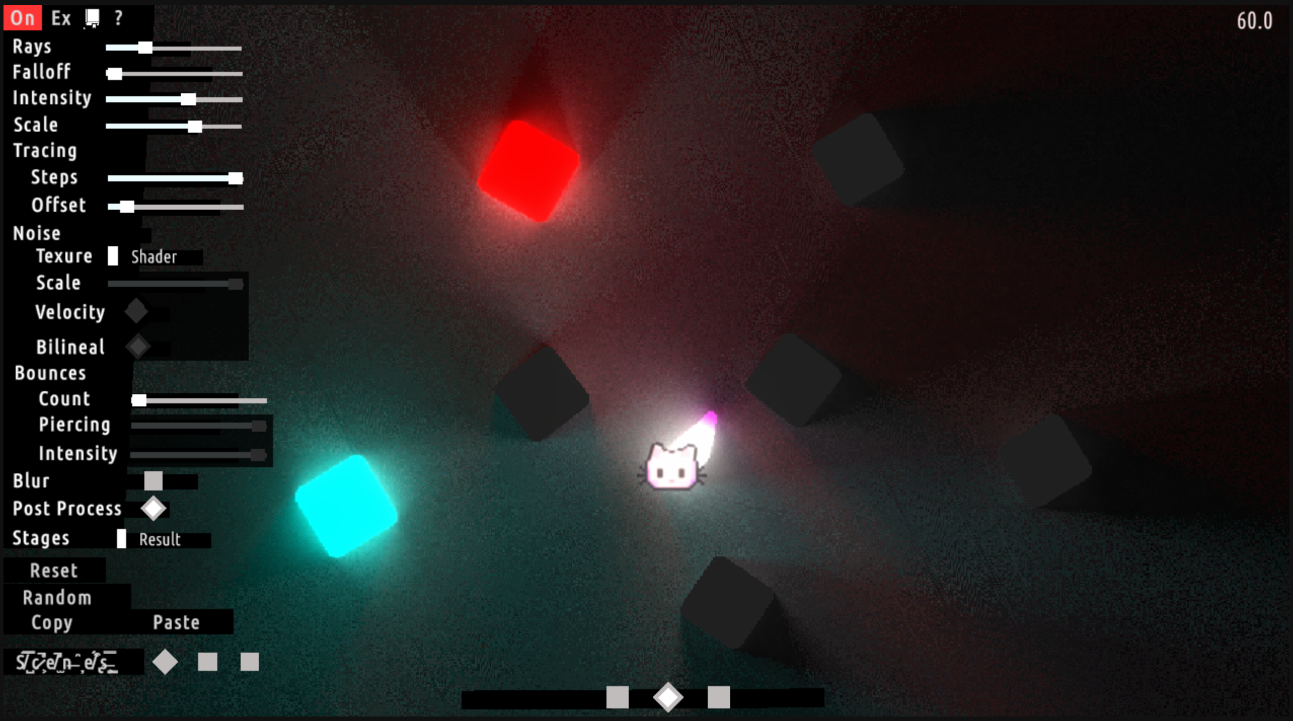
* GiLight2D contains quite a few options, the description of what each of them does can be seen in the tooltips.



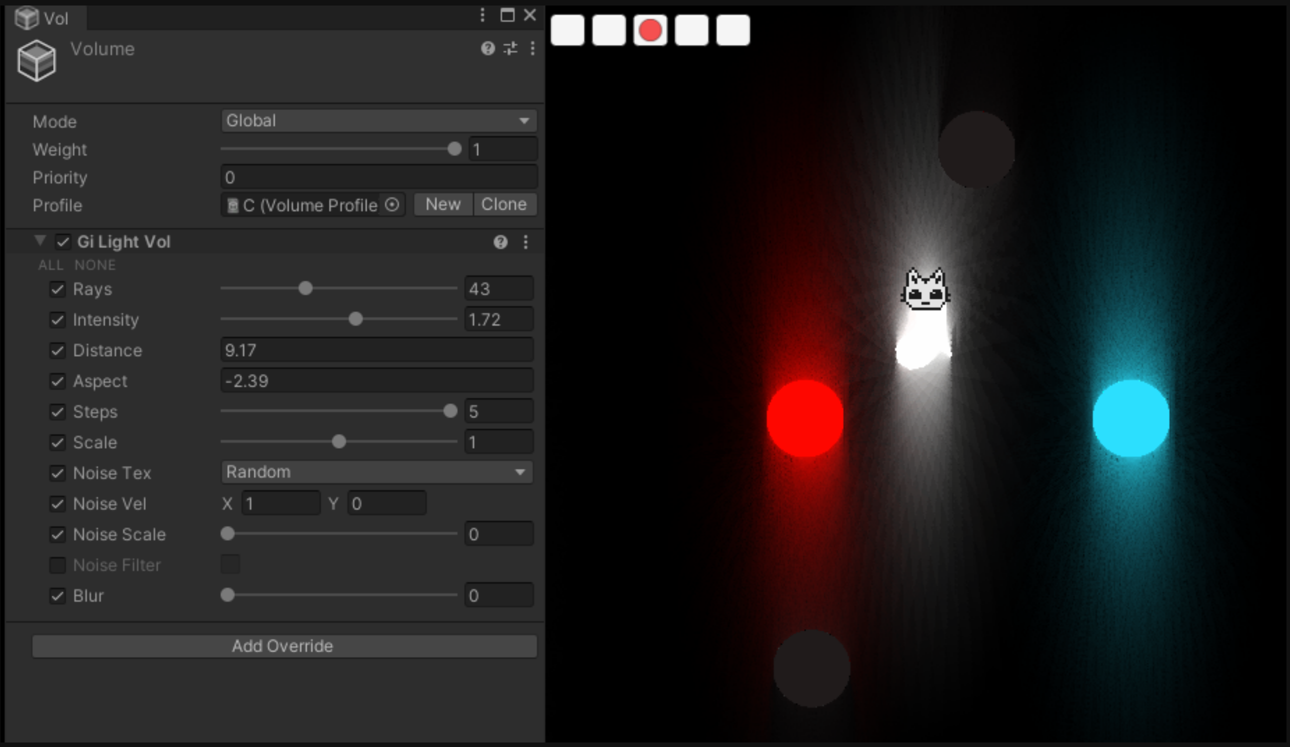
```
  Steps        N16 ▾
  Dist Offset  ●———————— 0
▶ Noise        Texture ▾
  Border       ●———————— 0
Additional orthographic camera space, to make
objects visible outside of the camera frame      1
```

Examples

Adjust and play with parameters you can in WebGl Demo.



Configured example contains in Project Samples. It uses Urp Asset with configured GiRenderFeature and outputs the result via shader as texture.



Vol

Volume

Mode	Global
Weight	1
Priority	0
Profile	C (Volume Profile)

New Clone

Gi Light Vol

ALL NONE

✓	Rays	43
✓	Intensity	1.72
✓	Distance	9.17
✓	Aspect	-2.39
✓	Steps	5
✓	Scale	1
✓	Noise Tex	Random
✓	Noise Vel	X 1 Y 0
✓	Noise Scale	0
	Noise Filter	
✓	Blur	0

Add Override