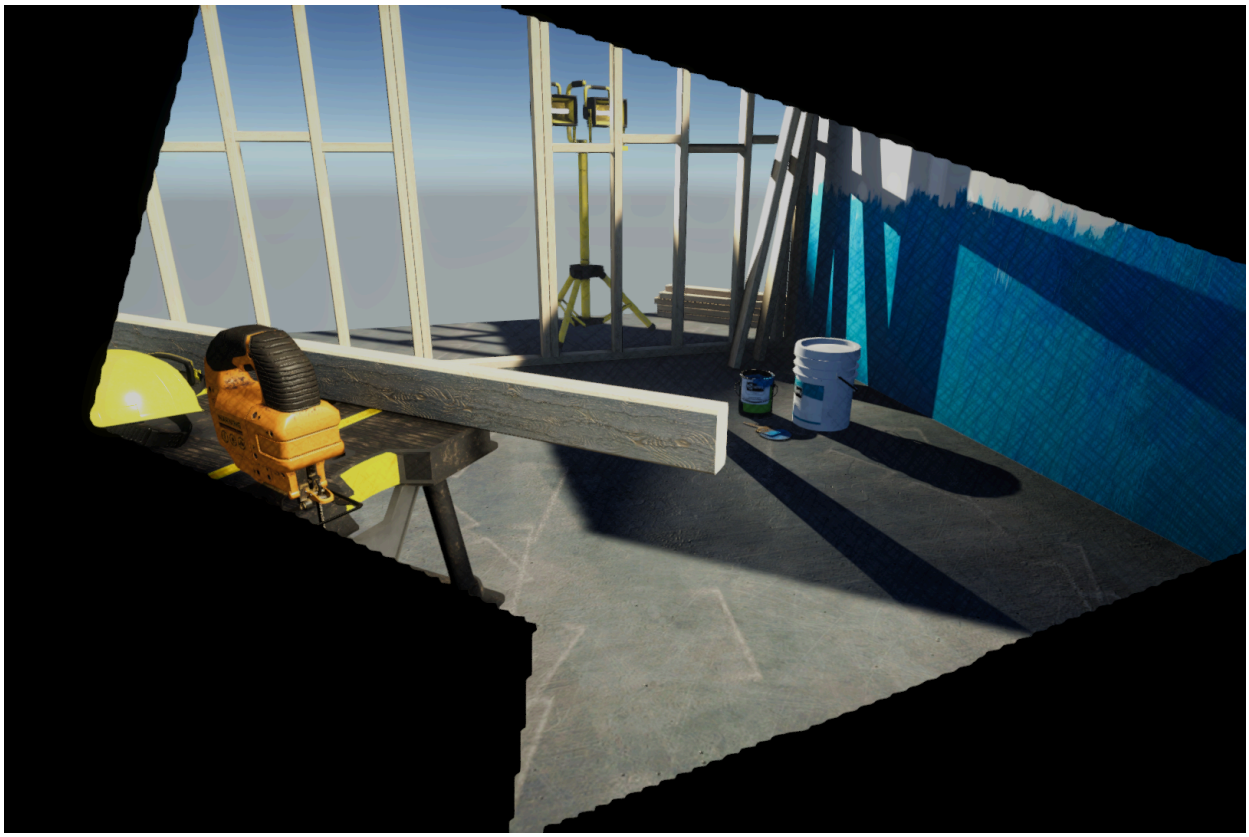


Transition Post Processing

By Joseph Y.

—

Overview

The transition post-processing effect is an image effect that creates screen transitions to a selected screen color (defaulted to black). This prevents the need to use animations or tweening of UI objects to do transitions. It is designed to run on the Unity URP pipeline and Unity 2019.3.9f1. This effect utilizes URP's render feature to integrate it into the render pipeline and ShaderGraph to create the shader effect.

The shader for this effect is based on Dan Moran's Shaders Case Study—Pokémon Battle Transitions (https://youtu.be/LnAoD7hgDxw?si=tCtTEOshaZdfLi6R). The shader was modified to remove the distortion ability, as we want it to be more of a general-use effect.

Installation

Before you begin:

1. Ensure you use the Universal Render Pipeline (URP) and Unity 2019.3.9f1.
2. Ensure you have installed the necessary packages, such as Post-Processing.
3. Ensure that you are using forward rendering in your universal renderer.
4. Ensure that the color space is set to linear in the project settings.

The effect is contained in a Unity package you can import into Unity. To do this, navigate to Assets > Import Packages, select the Unity package, and open it. A popup will appear showing the contents of the package. Choose to import everything from the package and click OK. They will be imported into the Unity project under the Post Processing folder/Transition folder.

After importing, you should see several specific items, including a folder named "textures" containing various transition textures, 3 C# scripts (1 for the render feature, 1 for the render pass, and another for the volume component), and the transition shader.

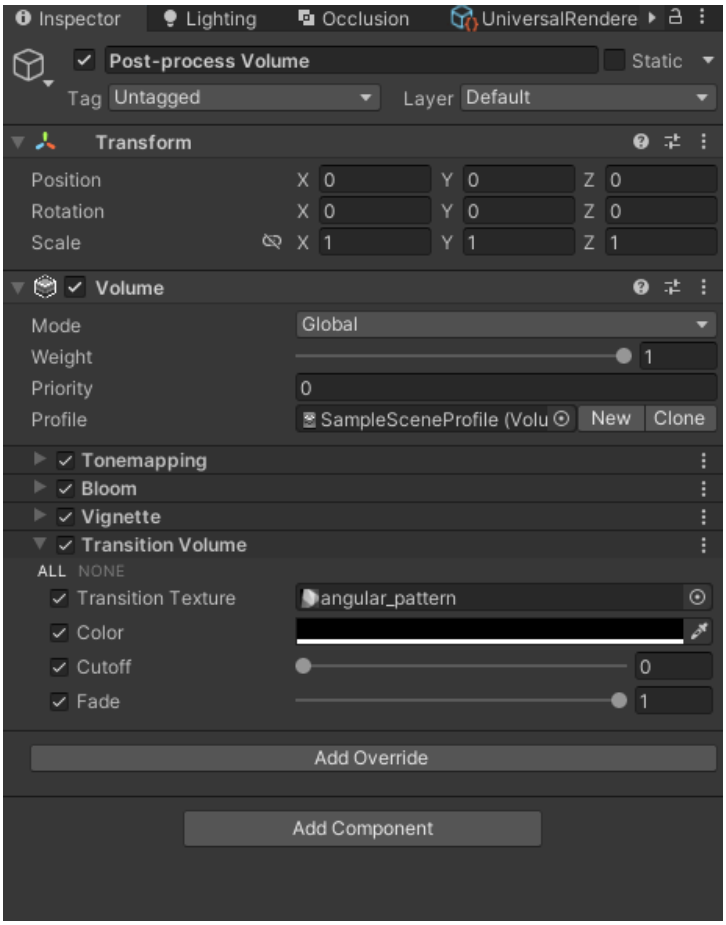
To apply the effect:

1. Navigate to the universal renderer being used in your project and locate the "Add Render Feature" button.
2. Click the "Add Render Feature" button and choose the transition render feature.
3. Create a game object or go to the object containing your post-processing volume/volume.

4. Under the volume component, click on **Add Override** and select Transition. If done correctly, the volume component should be updated with a new volume override that says Transition Volume.

You can adjust the render pass event by navigating to the Transition render feature in the universal renderer. Click on the dropdown beside the setting and change the event to your preference, such as "after opaques."

Using the transition effect



You can go to the volume game object with the transition effect to adjust it. You can use one of the provided transition textures in the textures folder or a custom one (refer to Transition Textures/Creating and Using custom textures). Here are the details about the parameters:

1. **Transition Texture:** The field containing our texture for the transition. For more info about the transition textures, refer to Transition Textures/Creating and Using custom textures.

2. **UseScreenTexture:** A parameter to enable the replacement of a screen color with a texture during the transition. Mainly used for applying the last frame of the previous scene or area as a screen texture, refer to Using screen texture instead of screen color.
3. **ScreenTexture:** The field containing our texture to replace what is supposed to be the screen color. If nothing is inputted to the field, it will output a plain white texture by default.
4. **Color:** This parameter affects the color at the end of the transition. Defaulted to black.
5. **Cutoff:** The parameter that affects the transition itself. 0 = the start of the transition and 1 = the end of the transition. To animate the effect, you will need to call the volume in a script and get the transition effect. Once you do so, you can lerp it from one value to another. Defaulted to 0.
6. **Fade:** Affects the color's alpha at the end of the transition. 1 = Opaque and 0 = Invisible. Defaulted to 1.

Transition Textures/Creating and Using custom textures

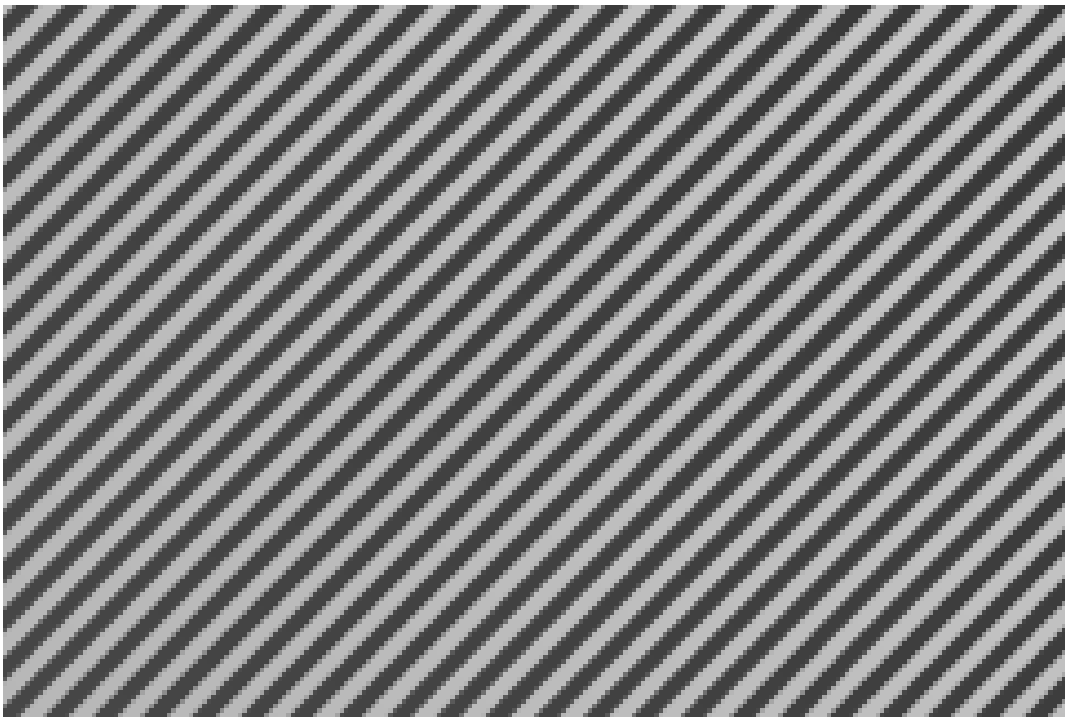
The transition post-processing includes a folder with different transition effects, such as an angular and a spiral transition. These textures are part of the Unity package from Dan Moran's Shaders Case Study - Pokémon Battle Transitions (https://youtu.be/LnAoD7hgDxw?si=tCtTEOshaZdfLi6R), which can be downloaded from the link provided in the video description.

The basic left to right transition texture

The textures are mostly in grayscale, with black representing the start of the transition and white representing the end. However, the diagonal distort and criss-cross pattern textures are red and green, unlike the others.

Diagonal Distort texture pattern

Even though it might not be obvious, when you grayscale it, the red turns black and the green turns white. This means that the red behaves like the beginning of the transition, while the green behaves like the end.



Diagonal Distort texture pattern but grayscale

In my opinion, it is recommended to use black-and-white or grayscale for your transition textures. In this case, black represents the start of the transition and white represents the end of the transition.

When you import the texture to your Unity project, there are certain settings to consider. One crucial setting to change is the wrap mode, which should be set to "clamp" instead of "repeat." If left on repeat, artifacts may appear on the edges of the screen during transitions.

If you look closely at the edges, you can see some gaps. It looks like this if you left it on repeat in the wrap mode. It can be avoided if you change it to clamp in wrap mode.

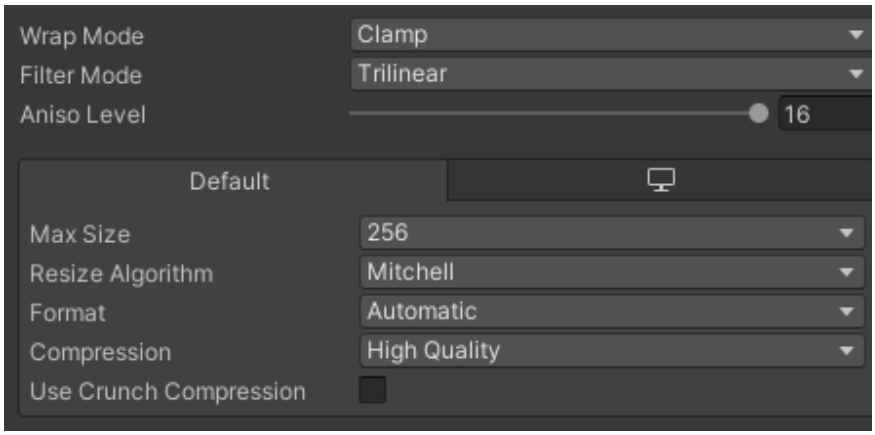
If you want smooth transition edges, use the trilinear or bilinear filter mode. For a slightly pixelated look, set the filter mode to point.

Point Filter Mode

Trilinear Filter Mode

Depending on the texture, you can adjust the texture compression level from none to high quality. This will impact the compression of the image in the build and project, as well as how the transition looks during a mid-effect. It's important to be mindful of this. If you're unsure, it's best to keep the compression at none, especially if the texture is already small enough to prevent it from becoming a significant concern.

The image below is the setting that I commonly use for the transition textures.



You don’t have to follow this strictly as different textures may have different needs or settings to be optimal.

You may need to use application software like Photoshop to create the textures. It is best if the texture is on a gradient from black to white, but different shades of black and white should also work well. Some good examples of this include:

Using screen texture instead of screen color

If you want to transition to a texture or a particular captured frame of the screen (via converting the screen capture as a texture) instead of a screen color (which is normally black by default), you can enable the parameter, **UseScreenTexture**, and a texture into the **Screen Texture** field to do so. Lerping the cutoff value from 0 to 1 will transition the screen to the chosen texture (or a plain white texture).

So in situations where for example, you want to transition from one scene to another but want the illusion that the player is somewhat still seeing the view of the old scene before being transitioned to the new scene, you will need to start your cutoff value at 1 and then lerp to 0. You would normally use the captured frame at the time you start the transition. You can either save it in a variable if it's within the same scene file or save the image in directory, read it in a script, convert and save it to a texture2D variable and then set the **Screen Texture** field with the Texture2D variable.

Update Log

V0.95: Ability to use a texture instead of a screen colour, optimizations & edits to code.

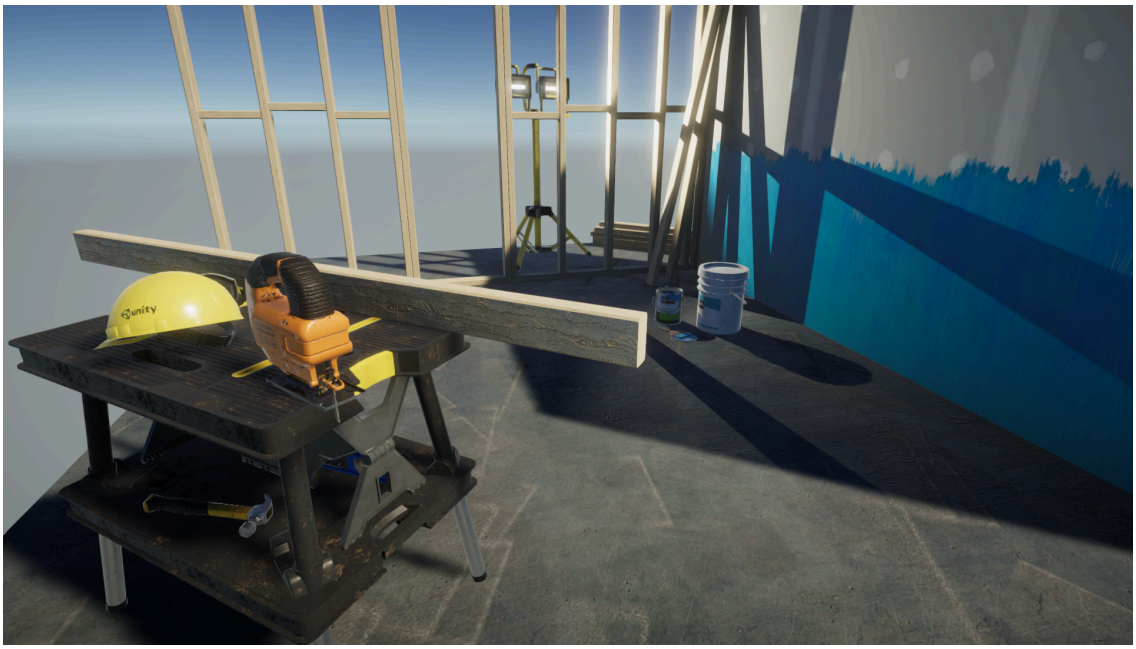
V0.9: The base implementation of the Transition post processing.

Credits

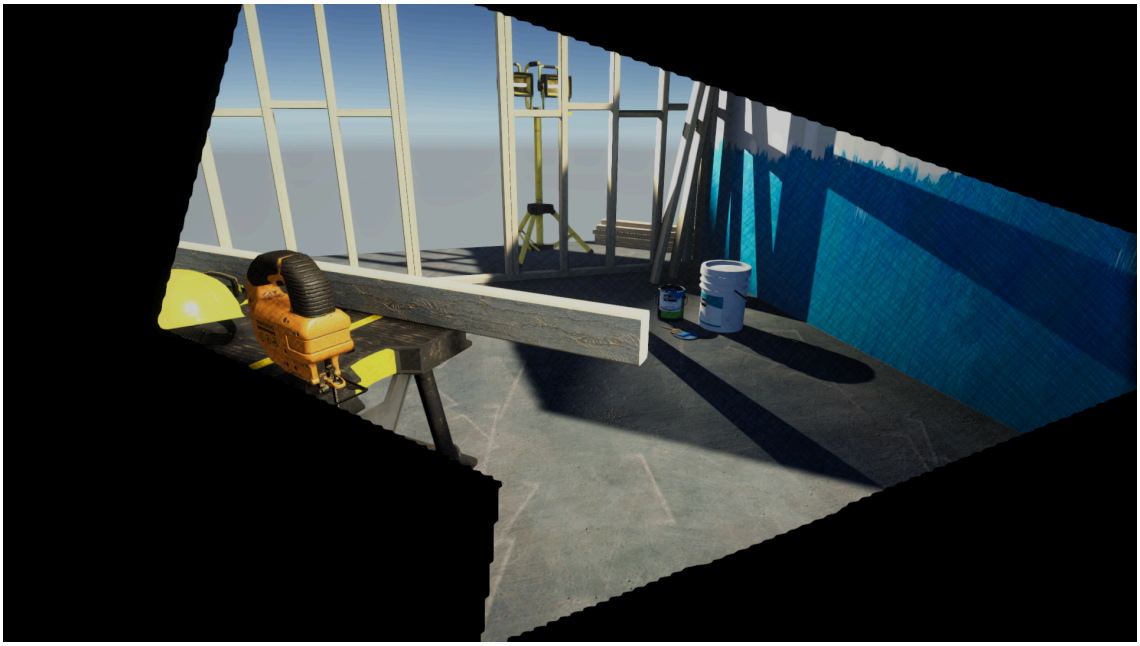
Shader code is based from Dan Moran's Shaders Case Study—Pokémon Battle Transitions (https://youtu.be/LnAoD7hgDxw?si=tCtTEOshaZdfLi6R) YouTube video.

The provided transition textures are made by Dan Moran from his Shaders Case Study—Pokémon Battle Transitions (https://youtu.be/LnAoD7hgDxw?si=tCtTEOshaZdfLi6R) YouTube video under the **CC By 4.0 Attribution 4.0 International** License.

Example Screenshots



Without the transition



Angular Transition

Spiral Transition

Top and Bottom Transition