Warp Image Effect - Post Processing Documentation

1. Introduction

This asset provides a post processing camera image effect that can be used directly out of the box. The effect displaces (“warps”) pixels on the final image using a flow map texture. It can be used for various effects; like black holes, reality-warping visual effects, or different custom effects for specific game mechanics. It can also be used on mobile devices with little performance impact.

The effect currently only works with the built-in render pipeline. Supporting URP and HDRP is in the roadmap, look forward to future updates.

2. Usage

Add the required components to the scene camera. You need to add 2 components, the base controller class and a helper class to modify specific properties. Helper classes help create different behaviours for the effect. The components are explained below. If you want the effect to work on UI elements, set the render mode of the canvas to “Screen Space - Camera”. The effect works on both perspective and ortographic cameras.

The material used for the effect can be found in *“Warp Image Effect/Resources/Warp Material”*. The flow texture used can be changed with another red-green texture if desired. There are also 4 different textures to use for the effect itself.

The shader will not be explained here in order not to clutter the documentation, but almost every line of code also has a comment explaining it. If desired, it can be modified to further customize the effect.

2.1 Components

- **WarpEffectController**
  This base class is responsible for creating the materials and applying the effect. The color of the effect texture and the warp strength can be controlled from the inspector. It needs a helper class to control the effect, so that custom controls

can be applied more easily. There are 3 distinct helper classes included in the asset to use as-is, or to act as a blueprint to generate different controls.

- **Warp_Stationary**
  This is a sample helper class that defines a behaviour. The effect can be moved by holding down the mouse button. The inner radius and thickness of the effect, as well as the rotation of the effect texture itself can be controlled from the inspector.

- **Warp_ExpandingWave**
  This is another sample helper class. It defines a behaviour that makes the effects expand like a wave and repeats it with the given intervals. The effect center can be moved around by clicking anywhere on the screen. The effect thickness, expand speed and wave interval can be modified from the inspector, as well as the rotation of the effect texture itself.

- **Warp_Pulse**
  This is another sample helper class. It defines a behaviour that makes the effect pulse in and out with given intervals. The effect center can be moved around by clicking anywhere on the screen. The effect thickness, maximum radius and the time to reach maximum radius can be modified from the inspector, as well as the rotation of the effect texture itself.

2.2 Chaining Multiple Effects

Helper classes currently access the base controller class with a GetComponent call. If you wish to create multiple effects (one pulse and one stationary, for example), you will need to modify the code so that each helper class has a reference to a specific controller class. Then attach a distinct controller class and a helper class for each distinct effect you want, and link the references in the scene. For example, for two distinct effects, the camera needs two base controller classes added (WarpEffectController) and any two helper classes that has a reference to these controller classes.

3. Demo Scenes

3 demo scenes with some objects and some text are included in the asset to try out the effect. Each scene has a different helper class configured.

If you have any questions or suggestions, please feel free to write a review on the Asset Store, contact me by e-mail or join our discord server. You can e-mail me at the address shown on the Asset Store page, or the e-mail address given below:

burak.bayboga@gmail.com