

Stylized 3D Depth of Field Usability Testing

Thank you for participating in the usability testing for Depth of Field algorithms in Flair. This manual is intended to guide you (the tester) through the testing process. Please first read through the manual and then proceed with the tasks following the instructions in Google Forms (linked at the end). If at any point you feel lost, return to this manual.

There are a total of 4 algorithms intended to be used.

1. Adaptive Bilateral Depth Filter
2. Polygonal Apertures Depth of Field Shader
3. Adaptive Recursive Depth of Field Filter
4. Circular Separable Convolution Depth of Field

All of the algorithms are tested in a 3D rendering software called Flair that is used together with Autodesk Maya.

Autodesk Maya

To open a scene navigate to File → Open scene and search for the intended scene you want to open.

To move around the viewport: Alt + Left-mouse button to rotate, Alt + Hold down the Mouse wheel to move up-down-right-left and scrolling the mouse wheel to zoom in-out.

Flair

Some key aspects to know when using Flair.

Flair uses a node/graph-based workflow. The visual effects (like DoF) are added via nodes in the processing graph.

To open a pre-saved graph, navigate to File → Open Document and in the window select the graph you wish to see.

All of the images are imported into Flair with Read nodes (Figure 1) or Import nodes (Figure 2). Read nodes use pre-rendered images for whom to apply different styles, while Import nodes use scenes that are rendered in the Maya viewport. Visual effects are applied through shader nodes which import pre-written GLSL code. These GLSL files also have parameters defined in them that will appear in the user interface of Flair. Modifying these parameters in the Flair interface can have different stylization effects on the render that happen in real-time.

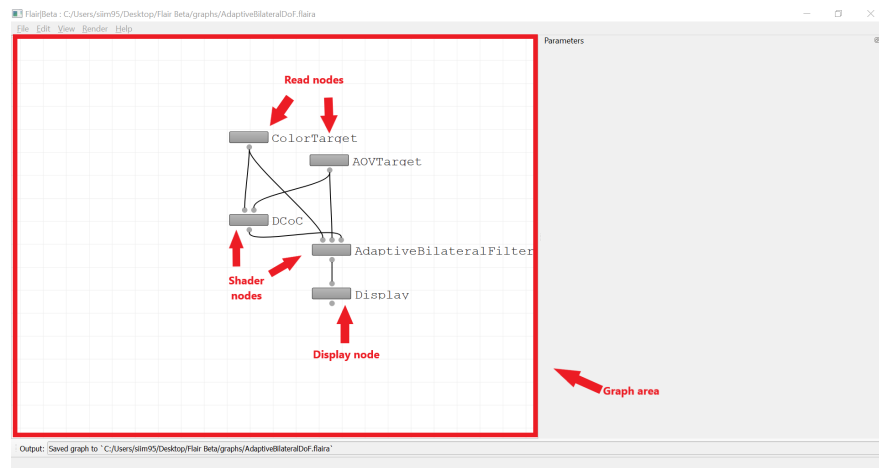


Figure 1. Read nodes

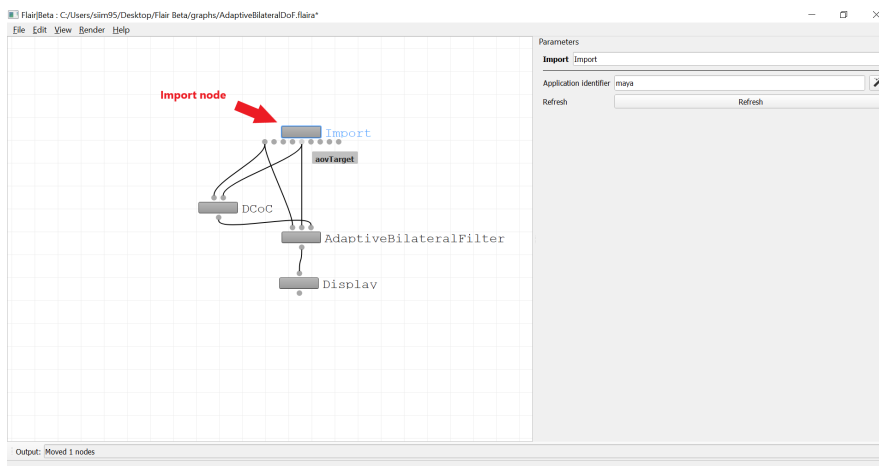


Figure 2. Import node

By right-clicking anywhere on the graph area a menu will appear that will allow you to place any of the aforementioned nodes in the graph (see Figure 3).

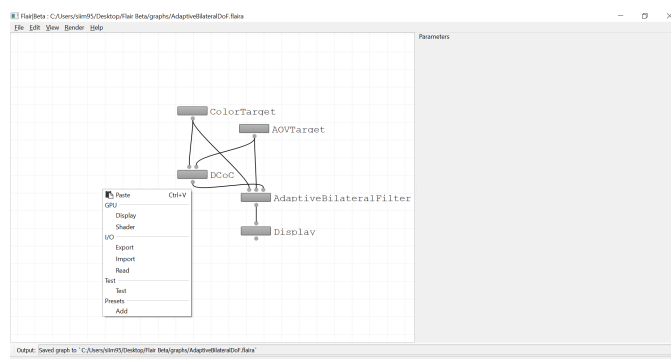


Figure 3. Graph menu.

When right-clicking on a shader node a menu will pop out. Clicking on the *Edit shader* will open a window from which to specify the GLSL code path and also to refresh the node (See

Figure 3). Refreshing should be done when loading a pre-existing graph or when changes have been made in the GLSL code.

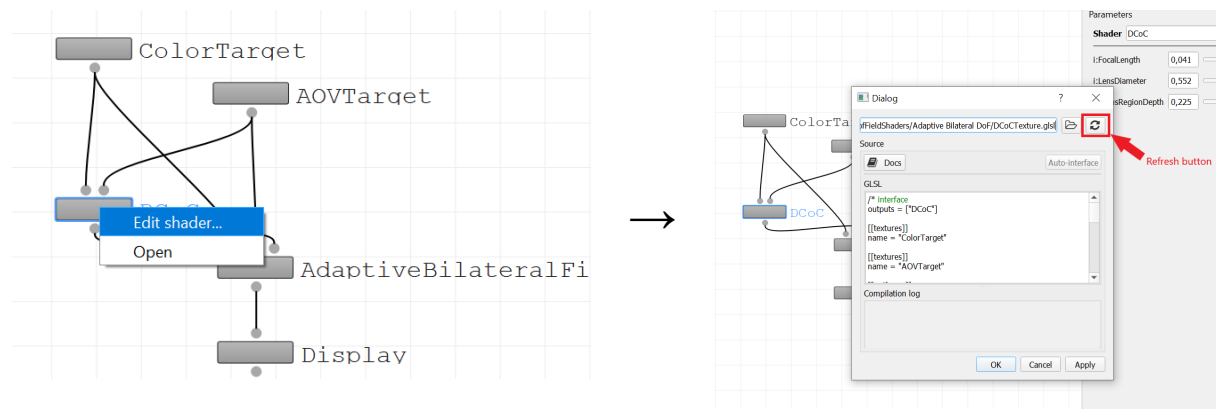


Figure 4. Shader node menu

To see the render applied on the scene or image a display node is needed first as seen on Figure 1. When right-clicking on the node a small menu will pop up. Clicking on the *Open display panel* will open the render output on the bottom right corner of the interface as seen on Figure 5.

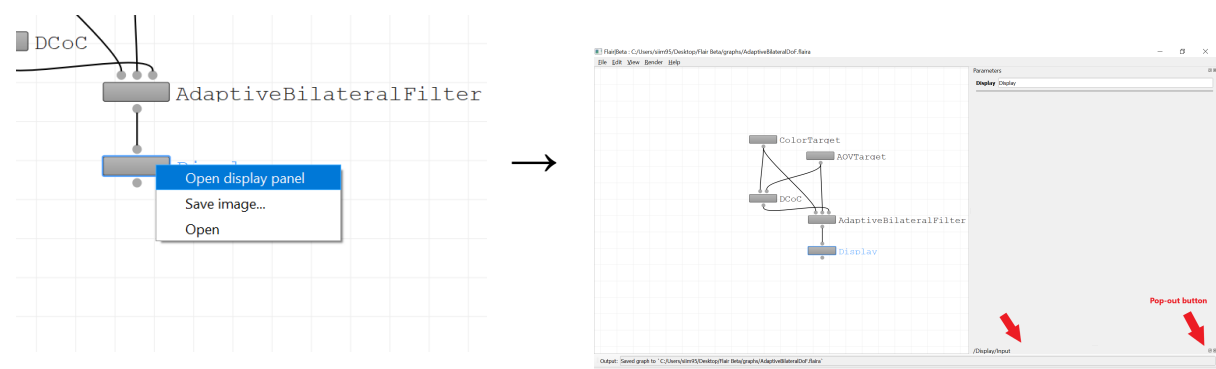


Figure 5. Opening the Display

By clicking the pop-out button the render can be seen. The render can also be seen simply by moving the cursor over to the Output section and then dragging it upwards as seen on Figure 5.

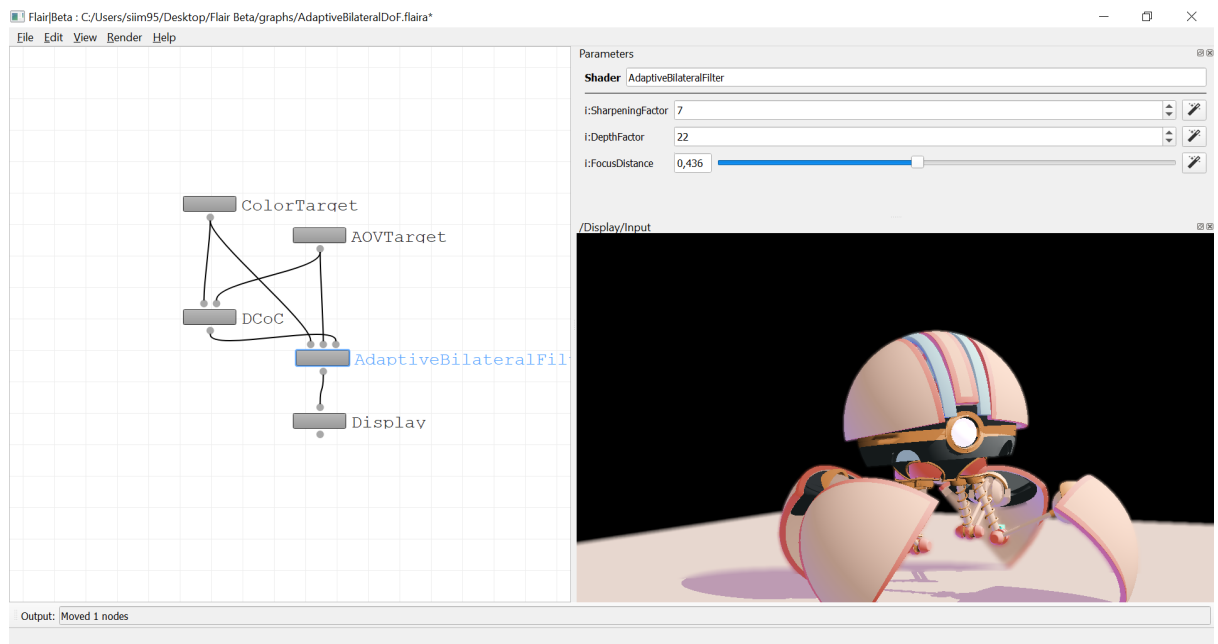


Figure 5. Render output

Testing

There are a total of 2 tasks needed to complete the testing.

Please click on the link to move on to the tasks.

https://docs.google.com/forms/d/e/1FAIpQLSf0mP7rqk05Lt-eAinZIdhxRubdEIVTSeNE5m2i8P2gTNoUoQ/viewform?usp=sf_link

Node and parameter explanations:

1. Adaptive Bilateral Depth Filter

1.1. DCoC - Calculates the circle of confusion for each pixel.

1.1.1. FocalLength - how strongly light is converged

1.1.2. LensDiameter - determines how much light is let in

1.1.3. FocusRegionDepth - blur distance

1.2. AdaptiveBilateralFilter - does the blurring

1.2.1. SharpeningFactor - how sharp or not the blurred area is

1.2.2. FocusDistance - move the blur area forward or back

2. Polygonal Apertures Depth of Field Shader

2.1. CoCmap - calculates the circle of confusion for each pixel

2.1.1. FocalLength - how strongly light is converged

2.1.2. FocalDistance - move the focus area

2.1.3. ApertureSize - determines how much light is let in

2.1.4. MaximumCoCDiameter - clamps the maximum CoC size

2.2. FirstPass - blurring first pass

2.2.1. Angle - determine the angle of the blur

2.3. SecondPass - blurring second pass

2.3.1. Angle - determine the angle of the blur

2.4. FinalPass - mixes together focused image and blurred image

3. Adaptive Recursive Depth of Field Filter

3.1. CoCMap - calculates the size of CoC for each pixel

3.1.1. Scale - blur scale factor

3.1.2. FocalLength - how strongly light is converged

3.2. BlurredDepthMap - a gaussian blur performed on a depth map

3.3. WeightMap - calculates the weight for each pixel

3.3.1. Cmin - determines the size of the focus area

3.3.2. FocalLength - how strongly light is converged

3.4. RecursiveFilter1-2 - blurring is done in multiple passes

4. Circular Separable Convolution Depth of Field

4.1. bufferA-D - needed for calculating blurring strength for each color channel

4.1.1. KernelSize - size of the kernel used to calculate final blurring

4.2. Image - adds together all the color channels

4.3. Composite - mixes together focused image and blurred image

4.3.1. Blend - how strongly the blurred image is overlapping the focused image

4.3.2. FocusDistance - how far the blurred region is

5. Stylized extension

5.1. toonShader - provides toon shading for the depth of field

5.1.1. lightDir - coordinates to modify the incoming light direction

5.1.2. Fraction - modify the distances of the shades the are on the bots

5.2. Warp - provides texture warping for the blurred area (Warp is in a donut shape)

5.2.1. Center - x and y coordinates to determine the center of the warp

5.2.2. distanceMult - determine how war the donut ring will reach from the center

5.2.3. shockParams - warp modifiers (starting modifiers should be [10.0 ; 0.8; 0.1])