Implementation of Post Processing Effects using Shaders

Finian J. Graham

Contents

[General Use of Post Processing Effects 2](#_Toc162267062)

[Implementation 3](#_Toc162267063)

[Gradient Tint 3](#_Toc162267064)

[HSL Colour Shift 3](#_Toc162267065)

[Basic Blur 4](#_Toc162267066)

[Underwater 5](#_Toc162267067)

[Two-Pass Gaussian Blur 6](#_Toc162267068)

[Retro 7](#_Toc162267069)

[Bloom 8](#_Toc162267070)

[Multi-effect implementation 9](#_Toc162267071)

[Polygon Area Effect 9](#_Toc162267072)

# General Use of Post Processing Effects

Post processing is utilized in both gameplay and artistic applications for a variety of different reasons. These reasons range from immersion, by making a scene look more realistic, to creativity, where a scene can be adjusted to take on a particular style.

An example of post processing to enhance immersion and realism is Bloom. Since a computer monitor can only reach a certain brightness, it can be tricky to truly emphasis how bright an object is. To do this, we utilize blooming to add a kind of “glow” to extremely bright objects.

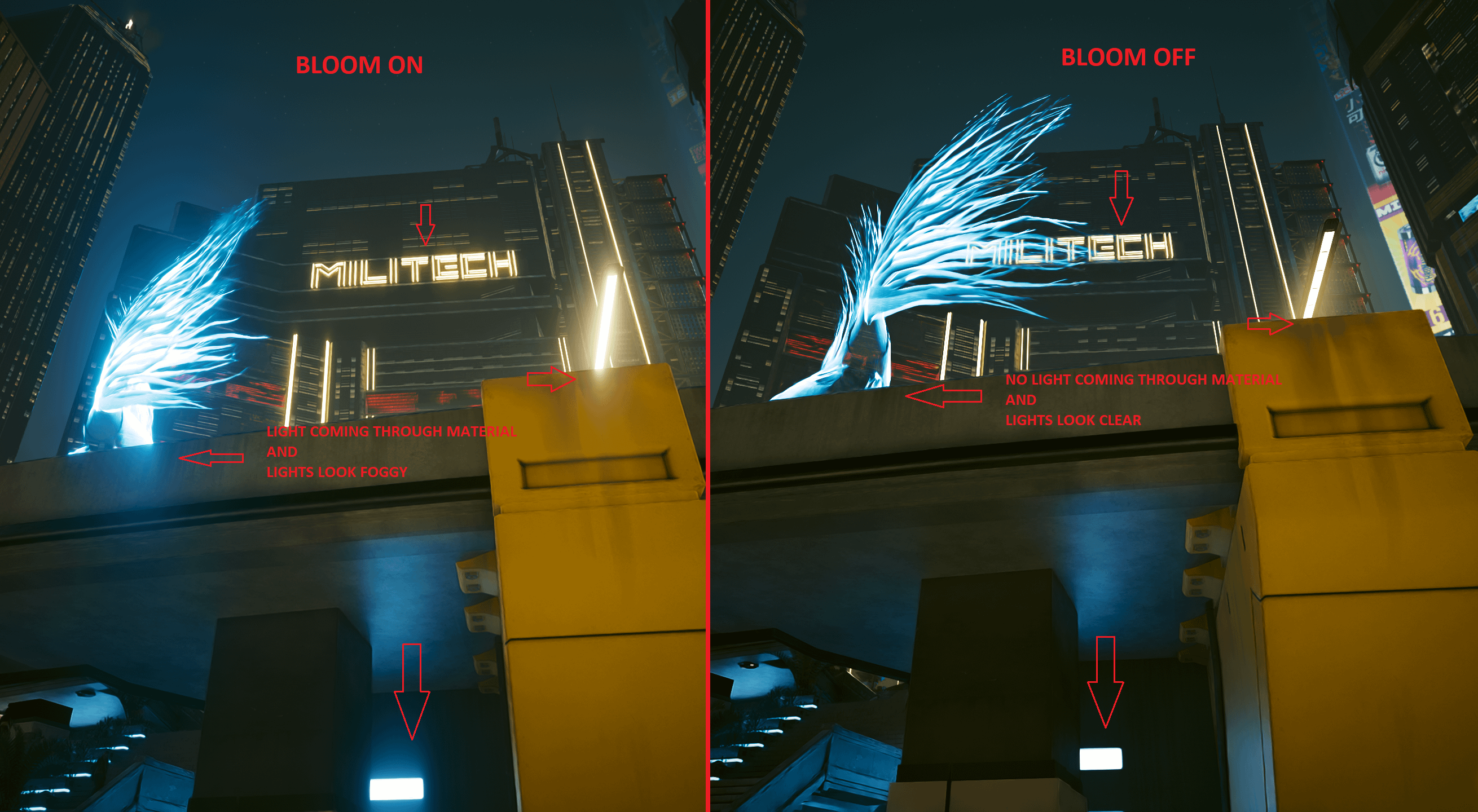


Figure 1: Bloom Example

(r/cyberpunkgame, 2023)

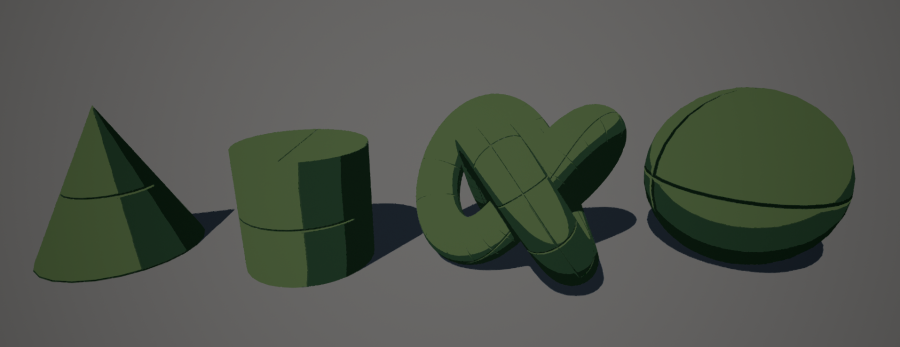
On the other hand, we can use “Cel Shading” as a form of artistic post processing to give a unique style to our game. Cel Shading converts lighting into a more banded transition, removing smooth blurry transitions in favour of a sharp contrasting line between darker and lighter sections of a model. 

Figure 2: Cel Shading Example

(Langmead, n.d.)

# Implementation

## Full Screen Effects

To create full screen post processing effects, the entire scene is rendered to a texture that can be modified. Once this texture is created, it is then drawn to a Quad that covers the entire viewport with the shader applied.

## Gradient Tint

Tinting the scene is simply done by multiplying the scene colour with a target colour. In order to achieve a gradient, two colours are supplied by the CPU and then transitioned between using a lerp based on how far down the screen the pixel is.

To improve the look of this gradient, a hue shift could be used instead.

A screenshot of a video game

Description automatically generated

Figure 3: Gradient Tint

### HSL Colour Shift

A screenshot of a computer program

Description automatically generatedTo make the two colours chosen for this gradient shift between different hues, a HSL to RGB conversion is used. The CPU can then increase the Hue of each colour over time before passing the new RGB values to the GPU.

Figure 4: HSL to RGB Function.

## Basic Blur

For the basic blur implementation, I utilized an Average Blur approach in which each pixel is set to the average value of the surrounding pixels, weighted to a mask. A benefit to this approach is that you can easily alter the effect by changing the mask, but it comes with the downside of being very slow if the mask is too large.

A group of boxes on a hill

Description automatically generated

Figure 5: Basic Average Blur

A bright light in the dark

Description automatically generated

Figure 6: Blur Mask

## Underwater

A screenshot of a video game

Description automatically generated

Figure 7: Underwater.

A close up of a blue and purple surface

Description automatically generated

Figure 8: Underwater Normal Map.

## Two-Pass Gaussian Blur

A screenshot of a video game

Description automatically generated

Figure 9: Gaussian Blur First Pass.

A screenshot of a video game

Description automatically generated

Figure 10: Gaussian Blur Second Pass.

## Retro

A video game screen with a few cubes and a light in the sky

Description automatically generated

Figure 11: Low Resolution Pass.

A pixelated video game of a video game

Description automatically generated

Figure 12: Colour Filter Pass.

## Bloom

A yellow and white object in the sky

Description automatically generated

Figure 13: Brightness Filter Pass.

A group of bright lights in the sky

Description automatically generated

Figure 14: Basic Blur On Brightness Mask.

A screenshot of a video game

Description automatically generated

Figure 15: Bloom.

## Multi-effect implementation

A blurry image of a building at night

Description automatically generated

Figure 16: Retro + Basic Blur

## Polygon Area Effect

A brick wall with a window

Description automatically generated

Figure 17: Localized Blur Through Window

# References

Langmead, A. (n.d.). *Cel Shading Post Process*. Retrieved from Github: https://nerivec.github.io/old-ue4-wiki/pages/cel-shading-post-process.html

*r/cyberpunkgame*. (2023, October). Retrieved from Reddit: https://www.reddit.com/r/cyberpunkgame/comments/16qyl0q/light\_bloom\_comparison\_on\_vs\_off/