Vive Foveated Rendering plugin for Unity Developer Guide

Version: 1.0

Contents

- Vive Foveated Rendering plugin for Unity Developer Guide
  - Introduction
  - System requirements
  - Principles and terminologies
  - Plugin usage
  - Performance Recommendations
  - Changes

Introduction

Vive Foveated Rendering is a rendering Unity plugin which reduces the rendering work load through cutting edge GPU technologies. This plugin supports both fixed and eye-tracked foveated rendering. The developer could easily apply foveated rendering to their VR application and adjust shading rate and region size for either better performance or better quality, according to their requirements.

System requirements

- **Target GPU architecture:** only works for GPUs with NVIDIA Turing™ architecture. For other GPUs, the plugin would not take effect.
- **Nvidia driver version:** release 430 and later
- **HMD with eye tracker:** Eye tracking only works for HTC Vive series HMDs with eye tracking capacity. For HMDs without eye tracking, the plugin automatically switches to fixed foveated rendering.
- **Target OS and Graphics API:** Windows DirectX 11
- **Unity version:** 2017.x or newer version

Principles and terminologies

Foveated rendering is a technique that synthesizes an image with less details outside of immediate focus of eye gaze. This would potentially provide GPU performance gain, a much needed feature for demanding virtual reality applications.

In this plugin, we divide user's field-of-view into 4 regions: **inner, middle, peripheral and outside regions** (from innermost to outermost; see the image below). Starting from the innermost region, the details are gradually reduced during the rendering process. The **size** and **shading rate** (which controls how much details are rendered) of each region could be tweaked by application developer.

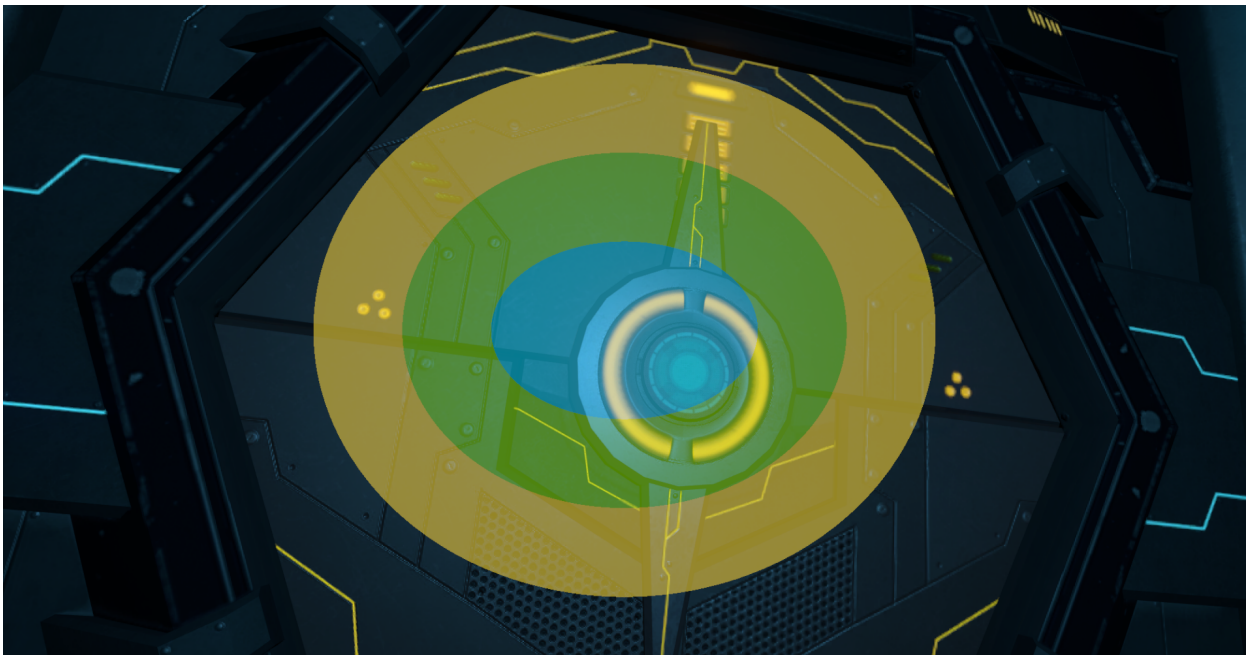


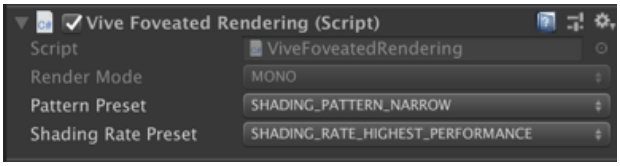
Fig. 1, HMD user's field-of-view divided into 4 regions.

Foveated rendering is not without its own shortcomings. One of the most noticeable artifacts is magnified aliasing in the peripheral region. This artifact is more obvious for thin or glossy objects in the scene. To ease this kind of artifact, the following options are our recommendations:

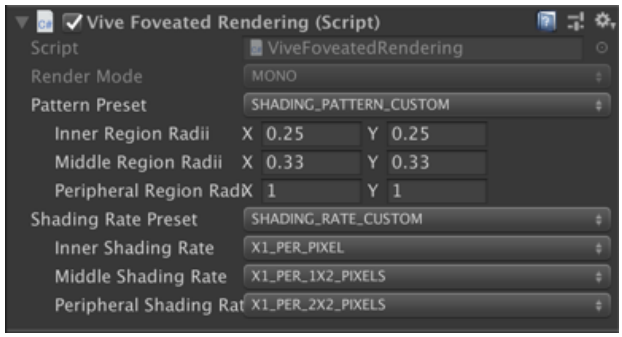
- Use temporal anti-aliasing. Unity's open-source implementation (https://github.com/Unity-Technologies/PostProcessing) is a good option.
- Tweak region parameters (size and resolution) based on the content in your scene. For scenes with a lot of text or glossy materials, developers should use a less aggressive setting.

Plugin usage

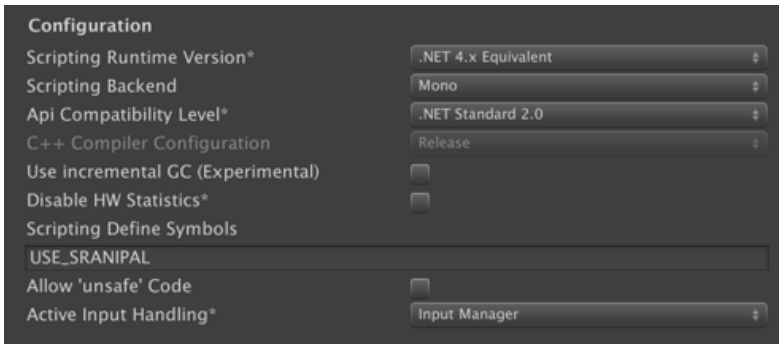
1. Import the Vive Foveated Rendering plugin.
2. Attach **ViveFoveatedRendering.cs** to your main VR camera.
3. Click play, the foveated rendering should be automatically enabled if your system is capable (see System requirements).
4. Region parameters could be tweaked from Unity inspector:



   - **VRS Render Mode:** This property would be automatically set by editor. Both "multi-pass" and "single-pass" are supported.
   - **Pattern Preset:** This property adjusts the region size of foveated rendering. There are several pre-defined region sizes: Narrow, Balanced, and Wide. The smaller more aggressive. There is also a "Custom" option (see point 5).
   - **Shading Rate Preset:** This property adjusts the shading rate of each region. There are several pre-defined shading rate: Highest Performance, High Performance, Balanced, High Quality, and Highest Quality. There is also a "Custom" option (see point 6).
5. Region size could be customized. Customization related fields would appear if you select "Custom" option. You could set the diameter of each region. The region out of peripheral region would be culled. Diameter of 1.0 equals to the width of single eye buffer in pixels.



6. Region shading rate could also be customized. Customization related fields would appear if you select "Custom" option. You could set the shading rate of each region:
    - CULL: nothing will be rendered
    - X1_PER_PIXEL: Each pixel is sampled once.
    - Supersampling options
        - Each pixel is sampled more than once, which would result in less aliasing than X1_PER_PIXEL (but will take more computing power).
            - X16_PER_PIXEL
            - X8_PER_PIXEL
            - X4_PER_PIXEL
            - X2_PER_PIXE
    - Coarse shading options
        - With coarse shading, a group of pixels is only sampled once, which would result in performance gain, but with less details rendered.
            - X1_PER_2X1_PIXELS: 2x1 pixel group
            - X1_PER_1X2_PIXELS: 1x2 pixel group
            - X1_PER_2X2_PIXELS: 2x2 pixel group
            - X1_PER_4X2_PIXELS: 4x2 pixel group
            - X1_PER_2X4_PIXELS: 2x4 pixel group
            - X1_PER_4X4_PIXELS: 4x4 pixel group
7. To work with SRanipal Unity plugin for eye tracking, please add the define "USE_SRANIPAL" to your project's Scripting Define Symbols.

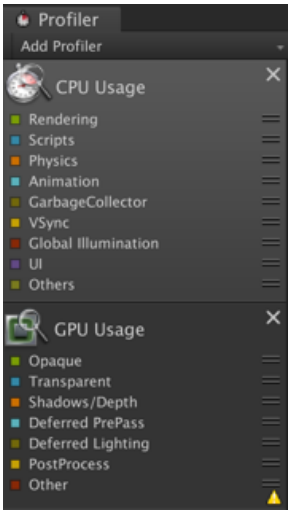


Performance Recommendations

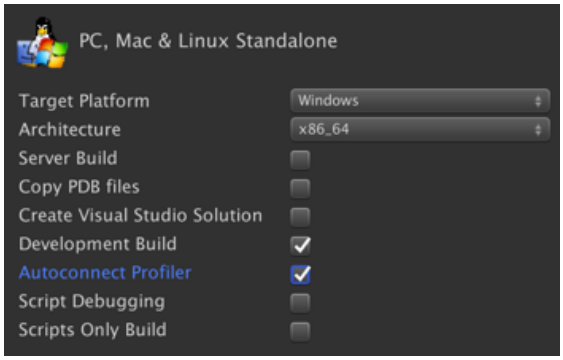
This plugin is based on NVIDIA **Variable Rate Shading** (VRS), which works by varying the numbers of pixels that can be processed by a single pixel shader operation. In other words, the work load of pixel shading could be eliminated. However, the performance could be impacted by many factors. The developer should diagnose their application to find the bottleneck for best use of foveated rendering.

1. Check profiler window for CPU and GPU loading.
    - Open profiler window: Window->Analysis->Profiler.

- Remember to add GPU profiler which could be found in profiler window->Add Profiler.



- Profiling the build could be better for analysis the performance. Check “Development Build” and “Autoconnect Profiler” in Build Settings. Then you could profile your build through profiling window.



2. Be warned of the overhead of over switching on and off foveated rendering.
3. Tuning custom patterns and regions for best performance and quality tradeoff.
4. The commands order inserted to camera could be adjusted to best fit the application if needed. Check this in **ViveFoveatedRendering.cs**.

Changes

v1.0.0

- First public release