Pacific Graphics (2021)
E. Eisemann, K. Singh, and F.-L Zhang (Guest Editors)

# Temporally Stable Content-Adaptive and Spatio-Temporal Shading Rate Assignment for Real-Time Applications

S. Stappen, J. Unterguggenberger, B. Kerbl and M. Wimmer

TU Wien, Institute of Visual Computing & Human-Centered Technology

**Abstract**

*We propose two novel methods to improve the efficiency and quality of real-time rendering applications: Texel differential-based content-adaptive shading (TDCAS) and spatio-temporally filtered adaptive shading (STeFAS). Utilizing Variable Rate Shading (VRS)—a hardware feature introduced with NVIDIA's Turing micro-architecture—and properties derived during rendering or Temporal Anti-Aliasing (TAA), our techniques adapt the resolution to improve the performance and quality of real-time applications. VRS enables different shading resolution for different regions of the screen during a single render pass. In contrast to other techniques, TDCAS and STeFAS have very little overhead for computing the shading rate. STeFAS enables up to 4x higher rendering resolutions for similar frame rates, or a performance increase of 4× at the same resolution.*

## 1. Introduction

With NVIDIA's Turing architecture, variable rate shading (VRS) allows devoting more processing power to regions of the screen that contain many details and devoting less processing power to other regions [NVI18]. The challenge is to determine which regions to shade in which resolution. We propose two techniques for controlling the shading rate: Texel Differential based Content-Adaptive Shading (TDCAS) and Spatio-Temporally Filtered Adaptive Shading (STeFAS). TDCAS uses the differentials of texture coordinates in $x$- and $y$-direction to determine regions that could be rendered with a lower shading rate. STeFAS utilizes the frame-to-frame color differences to detect over- and undersampled regions over time and adapts the shading rate accordingly. Previously undersampled regions are sampled at higher rates to reduce aliasing and improve quality. STeFAS does not depend on specific scene properties. Hence, it is generically applicable and offers higher quality than TDCAS while remaining almost as fast.

## 2. Texel Differential-Based Content-Adaptive Shading

Our content-adaptive shading algorithm strives to be generally applicable to many scenes while not introducing unnecessary aliasing. TDCAS selects the shading rate based on texel differentials, which are the partial derivatives in x and y direction of the texture space. We scale the derivatives by the inverse fragment size to take the influence of the current shading rate into account and by the inverse texture size to correlate the derivatives with texture pixel sizes. The full formula for the texture differential value is given in Equation 1.

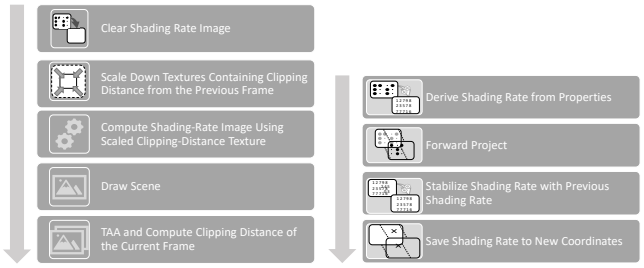
$$dt = \frac{|\frac{\partial u}{\partial x}| + |\frac{\partial u}{\partial y}|}{texSize.x \cdot fragSize.x} + \frac{|\frac{\partial v}{\partial x}| + |\frac{\partial v}{\partial y}|}{texSize.y \cdot fragSize.y} \quad (1)$$

Texel differentials are very valuable as they convey the information of the used mipmap-level of a texture. Larger values of texel differentials result in higher mipmap levels being selected. Depending on the screen resolution and the texture resolution, with very large texel differentials, whole textures might fall into a single pixel. Hence, the shading rate can be reduced for regions with high texel differentials. Very small texel differentials are an indication of an oversampled texture and the shading rate can therefore be reduced. Texel differential values between these two maxima can be sampled at the usual shading rate or higher shading rates. We use a shading-rate palette with ascending values, the higher the index the higher the shading rate. The transformation function deriving the index into the so defined shading-rate palette consists of three basic operations. First, the differentials are shifted from the range $[0,1]$ to $[-1,1]$. Second, a quadratic function is applied to take the quadratic space (x times y) into account and flip originally low values to high values, as both should be assigned low shading rates. Third, we subtract the result from 1 so both very high and very low differentials map to low indices and thus low shading rates.

## 3. Spatio-Temporally Filtered Adaptive Shading

A high-level overview of our STeFAS algorithm is given in Figure 1a (executed on the CPU) and Figure 1b (executed on the GPU). Our algorithm starts by clearing the shading-rate image to the highest shading rate to avoid artefacts from previous frames. In the next step, a color difference image to the previous frame is scaled down to $\frac{1}{16}$ of its resolution in x and y direction to obtain an average color difference for each 16x16 tile. Afterwards, a compute shader uses this scaled-down color-difference image to compute the shading-rate image, which is set up for the VRS pipeline and used during

all subsequent draw calls. If TAA is part of an application's render loop, color difference textures can be shared among the TAA step and STeFAS—which means that STeFAS' overall costs are diminished for such setups—otherwise color differences and clipping distances [Kar14] must be generated explicitly for STeFAS. The compute shader derives a shading rate for each 16x16 pixels tile by dividing the color-difference space in equidistant regions, each assigned a shading rate. Larger differences are assigned higher shading rates and smaller differences are assigned coarser shading rates. As we compute the shading rate from the previous frame's properties, we have to do a forward projection to get the new screen-space coordinates. This is possible by computing the world coordinates of the previous frame using a backward projection. These world coordinates are then transformed to the screen-space coordinates of the current frame and used to fetch the previous shading rate. In the last step, the derived shading rate is saved at the new coordinates. Regions that moved or are not hit by the forward projection due to perspective distortions are handled by the default shading rate, set in the first step of the algorithm as the maximum shading rate.




**(a)** *The CPU-side steps of STeFAS.* **(b)** *STeFAS steps in compute shader.*

**Figure 1:** *Overview of the STeFAS algorithm*

## 4. Preliminary Results and Conclusion

We capture the average frame times of animations for various configurations in two test scenes for performance evaluation. To analyze the quality of our approach, we compute the SSIM and the PSNR of frames at fixed positions of these animations. The scene used during the evaluation is a modified version of Crytek's Sponza. We performed the evaluation at various combinations of resolution and MSAA sampling count. Resolutions include 1080p, 2560x1440, 4k and 8k to cover potential future workloads. Sampling count configurations range from 1× to 8× MSAA. Figure 2 shows the performance in frame times and the SSIM in percentage. Ground-truth images for the SSIM and PSNR computations were rendered at 8k resolution with 8× MSAA and VRS disabled.

Our techniques are at least 33% faster in configurations utilizing higher sample rates and focusing on quality, e.g. all evaluation scenarios using 8× MSAA. In the performance-wise best cases, utilizing only coarse shading rates, e.g., all evaluation scenarios using 1x MSAA, we achieve four times faster frame computations. Alternatively, our techniques enable the rendering at higher resolutions, e.g. four times the resolution in amount of pixels, at the same performance levels of rendering at the lower resolution. For example, rendering with a resolution of 3840x2160, our VRS-enabled rendering is still 5% faster than rendering at a resolution of 1920x1080 without VRS. This enables higher resolutions while keeping the same frame rates. In terms of quality, the configurations focussing on performance deliver SSIM values with a deviation of 5% - 11%.

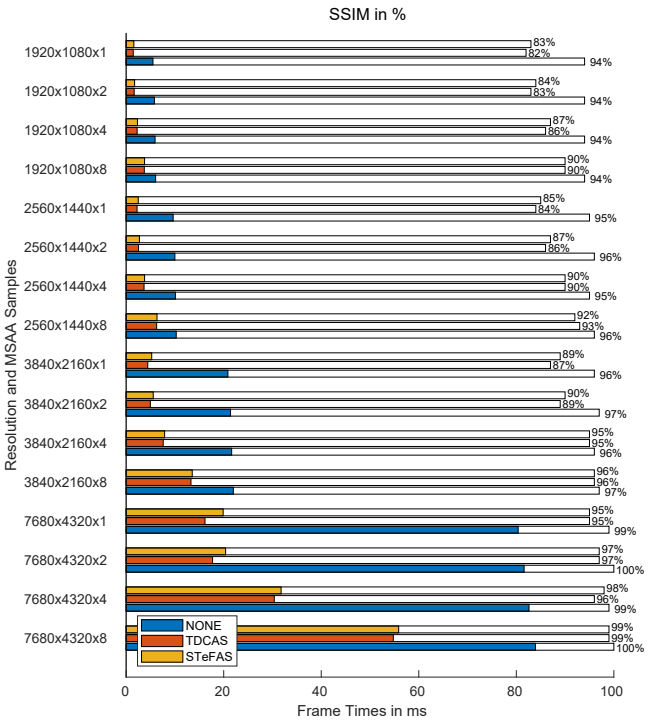



**Figure 2:** *Performance in frame times and quality in SSIM. Filled color boxes represent performance, empty boxes SSIM values.*

Remaining challenges to tackle include advanced temporal stabilization techniques for the shading rate and the rendered image. Further improvements in TAA are possible, like increasing the weight for previous frames. Moving objects, however, pose further difficulties which can lead to ghosting and noticeable artifacts. We would like to investigate combinations of TDCAS and STeFAS, and expect further performance and quality improvements. Simple approaches select the minimum/maximum shading rates to focus on performance/quality, respectively. More advanced approaches may strive to combine the strengths of both techniques.

## Acknowledgements

This work was supported by the Research Cluster "Smart Communities and Technologies (Smart CT)" at TU Wien.

## References

[Kar14] KARIS, BRIAN. "High-quality temporal supersampling". *Advances in Real-Time Rendering in Games, SIGGRAPH Courses* 1 (2014), 1–55.

[NVI18] NVIDIA. *NVIDIA Turing GPU Architecture*. https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/technologies/turing-architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf, accessed on 12.11.2019. NVIDIA Corporation. 2018.