Spatio Temporally-Filtered Adaptive Shading and a Generalized Shading Rate Stabilization Technique

Stefan Stappen*
Supervised by: Johannes Unterguggenberger†

Institute of Visual Computing & Human-Centered Technology
Research Unit of Computer Graphics
Vienna University of Technology
Vienna / Austria

Abstract

We propose a novel method to improve the efficiency and quality of real-time rendering applications, Spatio Temporally-Filtered Adaptive Shading (STeFAS). Utilizing Variable Rate Shading (VRS), a new hardware feature introduced with NVIDIA Turing graphics card micro-architecture released in 2018, and properties derived during Temporal Anti-Aliasing (TAA), our technique adapts the resolution to improve the performance and quality of real-time applications. VRS enables this kind of adaptation of the shading resolution throughout the screen during a single render pass. Our technique examines the color differences over time to find a fitting shading resolution. With each frame, this shading resolution is refined and adapted to the color differences accordingly. In contrast other techniques, STeFAS has very little overhead for computing the shading rate. This is one of the reasons why STeFAS enables up to 4x higher rendering resolutions while keeping up the same performance or 4x better performance at the same resolution. One problem with VRS is that varying shading rates may introduce flickering artefacts into real-time rendering applications. To alleviate these artefacts we focus on their root cause and propose a technique that stabilizes the shading rate. Our technique delivers more stable shading rates and reduces such artefacts significantly. In some cases, it is able to completely eliminate them.

Keywords: Variable Rate Shading, Temporal Anti-Aliasing, Stable Shading Rates, Real-Time Rendering

1 Introduction

There are high, opposing requirements for real-time 3D rendering applications. While scenes should be visually appealing and often realistic looking, frame rates of at least 60 Hz for desktop applications, 90 Hz per eye for VR applications have to be maintained. An additional factor are the increasing native resolutions of new devices, enabling the presentation of more details and images of higher quality but at an increased performance cost because more pixel color values need to be computed.

NVIDIA's Turing Architecture comes with a new tool, Variable Rate Shading (VRS), to improve the performance and quality of real-time rendering applications in an elegant way. VRS allows to devote more processing power to regions of the screen containing high detail features and reducing the invested processing power of other regions, showing little visual detail. This is achieved by an adaptive resolution of the shading, the so-called *shading rate*. An increase in resolution is enabled by supersampling shading rates, and a decrease with coarse shading rates. With VRS the screen is divided into a grid with a tile size of 16x16 pixels and for each of these tiles a different shading rate can be specified. This specification is done by a shading-rate image of the size of $\frac{1}{16}$ in width and height of the target resolution.

The challenge is to determine, which regions should be rendered with a higher resolution and for which other regions a lower resolution is sufficient. We propose Spatio Temporally-Filtered Adaptive Shading (STeFAS), a technique for controlling the shading rate, achieving our set goal of being generically applicable, so does not depend on specific scene properties, delivers images of almost indistinguishable quality for many scenes and improves the performance. By applying both supersampling and coarse shading, our technique can improve the performance of real-time rendering applications while producing images of the same quality as without our technique or improve the quality while keeping the same performance. STeFAS utilizes the frame-to-frame color differences to detect over- and undersampled regions over time and adapts the shading rate accordingly. In combination with an adapted Temporal-Anti Aliasing (TAA) approach, previously undersampled regions are sampled at higher rates to reduce aliasing and improve quality, while previously oversampled regions are rendered with a reduced shading rate to save performance. An example with the shading rate overlayed is shown in Figure 1.

Similarly to Content-Adaptive Shading algorithms,

*stefan.stappen@cg.tuwien.ac.at
†unterguggenberger@cg.tuwien.ac.at

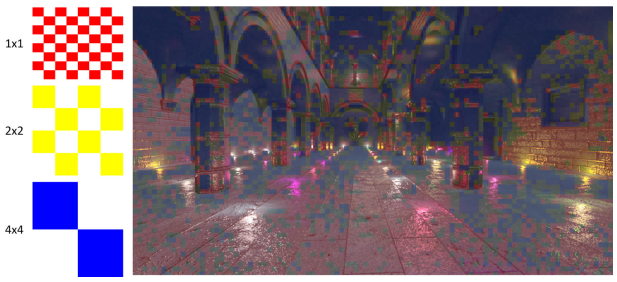


Figure 1: Example of our STeFAS technique with the effective shading rate displayed as an overlay. On the left-hand side the legend ting the shading resolution and the color code, is shown.

which are adapting the shading rate based on properties present in the content of the scene, STeFAS faces temporal instabilities in the shading rate. These instabilities can lead to artefacts of flickering colors, grabbing the attention of the viewer. We propose a novel approach to efficiently reduce these instabilities. For some scenes, we are even able eliminate such flickering artefacts completely. By applying a modified version of TAA on the shading-rate image itself, we tackle the problem at its root cause and are able to stabilize the shading rate.

In summary, our contributions consist of:

- A novel approach for controlling the shading rate: STeFAS, a technique that derives the shading rate from temporal color differences.
- A shading rate stabilization method that is independent from the mechanism that controls the shading rate. It is able to reduce temporal artefacts introduced by VRS methods susceptible to unstable shading rates.
- An efficient algorithm for VRS with STeFAS that is easy to integrate into an existing pipeline. It includes the steps for computing the shading rate image and post-processing with an adapted TAA approach.

In Section 2 the background of VRS will be discussed. Techniques based on VRS and TAA will be addressed, too. In the Sections 3 and 4 we explain the theory of our novel approach for controlling the shading rate and how our method for stabilizing the shading rate works. In Section 5 we detail the complete algorithm for integrating our techniques and VRS with an existing rendering engine. Finally, in Section 6 we discuss the achieved results and conclude with Section 7.

2 Background

Ramamoorthi et al. [6] lays the theoretical groundwork for reducing the shading resolution. They performs a gradient analysis for lightning, shading and shadows and are able to reduce the sampling while still preserving most of the details of the scene.

He et al. [1] presented with their multi-rate shading a concept that is pretty close to VRS. They state that fragment shading computations use up to 95% of the computational power for rendering a frame in real-time graphics. This highlights why techniques like VRS are advantageous for the field of real-time computer graphics: It enables better control of the area where we have most to gain. The extended graphics pipeline proposed by He et al., reduces the fragment shader computations per-pixel to a factor of two to three. In comparison to VRS, multi-rate shading performs rasterization on coarse fragments and then decides in coarse fragment shading if a fine per-pixel fragment shading is necessary. This leads to jagged images and more aliasing as edges are effectively rasterized at a lower resolution.

Ragan-Kelley et al. [5] propose extensions for modern graphics pipeline architectures that decouple visibility samples from shading samples. This is achieved by applying a hash function after each visibility sample which maps to a shading sample. The shading sample is cached and reused for further visibility samples. With this hash function, a many-to-one relationship between visibility and shading is defined. This extension would allow efficient supersampling for motion blur, defocus blur and also enables VRS. For example, with the hash function, one could reduce the shading rate by mapping four visibility samples to the same shading sample, or apply supersampling for visibility while keeping the shading rate at fragment size, as Multisample anti-aliasing (MSAA) does. With a hash-function, the shading rate can be configured for dynamic morphologies of dynamic sizes and does not have to be quadratic tiles of predefined size. This flexibility comes with the disadvantage that the hash function has to be evaluated for each visibility sample and is not hard-wired as with VRS.

Very close to VRS is the proposed adaptation of the graphics pipeline by Vaidyanathan et al. [7]. Like the extensions developed by Ragan-Kelley et al. [5], their method decouples the visibility samples from the shading. However, they already suggest a division of the screen into tiles and define the shading rate for each tile. The shading rate can be controlled either as interpolated values of vertex shader outputs, set as a constant value for the whole frame or as a radial function in screen coordinates for foveated rendering. The so defined coarse pixel size is then quantized to the closest predefined shading rate and the minimal size of all defined shading rates is taken as size for a tile to meet the required shading rate. In contrast, with VRS one choose from a predefined set of shading rates and the shading rate is controlled by a shading-rate image. Vaidyanathan et al. [7] also compensate for visible shading-rate changes by scaling the texture LOD with the coarse pixel size. This is handled similarly with VRS: The hardware automatically chooses the right texture LOD level based on the shading rate.

Due the novelty of VRS as a hardware feature for controlling the shading rate of real-time rasterization renderers, there are few techniques utilizing it. Yang et al. [11] proposes such a VRS technique that uses an error estimate based on spatial and frequency analysis to specify the shading rate. They estimate the error on a per-tile basis for half and quarter shading rates. The thresholds of the error estimate used to specify the shading rate is based on a perceptual model using average luminance per tile, environment luminance and a user specified parameter. Their technique delivers 5 % - 20 % total frame time reduction while pertaining a quality of above 90 % Structural Similarity Index Measurement (SSIM). However, the performance increase and quality are heavily dependent on the selected threshold parameter. With a higher threshold the performance increases but the shading rate becomes unstable. These shading-rate instabilities introduce temporal artefacts grabbing the focus of the viewer. Their workaround for this problem is to introduce two thresholds to mitigate the effects if the predicted error increases or decreases from the previous frame.

Another option would be to mitigate the temporal artefacts by post-processing effects, for example with anti-aliasing techniques. The current state-of-the-art for anti-aliasing techniques considering temporal effects, is temporal supersampling anti-aliasing (TAA) and was developed by Karis [3]. In contrast to preceding methods, like the one of Yang et al. [10], TAA only uses the previously resolved buffer for blending with the current frame, instead of multiple subpixel buffers. The detailed process can be viewed in Figure 2. In order to avoid clustering in space or time, a low discrepancy progressive sequence is used for the subpixel offsets, e.g, Halton sampling. Misses due to re-projection, which would result in ghosting, are handled by clipping the history value to the value of the current frame. This clipping is performed not in Chroma but Luma color space, as Luma has a high local contrast compared to Chroma. TAA delivers high-quality anti-aliasing with subpixel features at real-time ready performance rates. Nevertheless, it requires fine-tuning for the alpha and clamping parameters to avoid ghosting and other artefacts. Additionally, a static scene with no moving camera might result in flickering due to clipping.

Xiao et al. [9] combines coarse pixel shading with TAA, utilizing a novel scheme for jittering the samples over time to achieve a better distribution for shading and visibility than with Halton sampling. Their approach profits from the benefits of coarse pixel shading while mitigating the reduction in quality introduced by it.

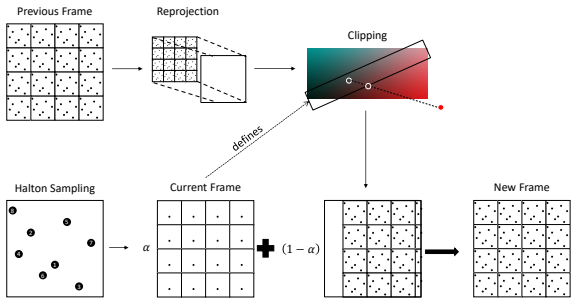


Figure 2: The main steps of TAA. Left in the bottom row, the current frame is sampled at the next Halton sample. At the top row, the previous frame is reprojected onto the current frame. Next, still in the top row, a clipping space based on the color values of the current frame is computed and used to clip the luma values of the reprojected frame. This image is then combined with the current frame by the weight α. The result of this weighted average then gives the anti-aliased new frame. Adapted from Karis [3].

3 Spatio Temporally-Filtered Adaptive Shading

3.1 TAA

TAA as described in Section 2 represents super sampling over time. By altering the position slightly each frame and blending pixels of previous frames with the current frame a higher sampling rate is achieved and it is therefore an anti-aliasing method.

To take motion into account, TAA computes motion vectors pointing from the current frame to the previous frame. These motion vectors are then used for a backward projection to get the correct position inside the previous frame. Due to perspective distortions and moving objects these motion vectors are not always accurate and artefacts of previously moving objects could be seen when their color values are combined with the current color value and their position has changed. These artefacts are called *ghosting*. Karis [3] proposed a technique to reduce these ghosting artefacts: clipping. By clipping the color values of the previous frame before they are combined with current color values, deviating color values, like they would occur in ghosting, are avoided. The clipping factors initially proposed are in a range from the minimum to the maximum values of the neighbourhood of the pixels of the current frame.

The problem with clipping is that it not only reduces ghosting artefacts but clips deviating color values in general. If there is missing subpixel information due to undersampling, subpixels of different frames can deviate heavily. Instead of combining the color values of these subpixels to get an average of them, with clipping the subpixel information from the previous frame is dropped in

the next frame. This effect is comparable to an instantaneous impulse and introduces flickering artefacts as illustrated in Figure 3. In this example there are four subpixels. Three have similar dark color values and the fourth is very bright. We first sample the top left subpixel and receive a dark color value. Lets assume the previous color value was dark too, so no clipping is performed. In the next frame, we sample the deviating, bright subpixel. The previous color value will be clipped to a range around this bright color value. The current bright color value will be combined with the previous, now brighter due to clipping, color value. In the next frame, we again sample one of the three similar dark color values. The bright previous color value will be clipped to the current color value and we will receive a combined darker color value. Each time, we hit the brighter color value we can observe two transitions from dark to bright and back to dark. The color is flickering. Coarse shading amplifies the problem because with coarse pixels spanning multiple pixels, the size of subpixels is as large as pixels and the chance of deviating color values is higher.

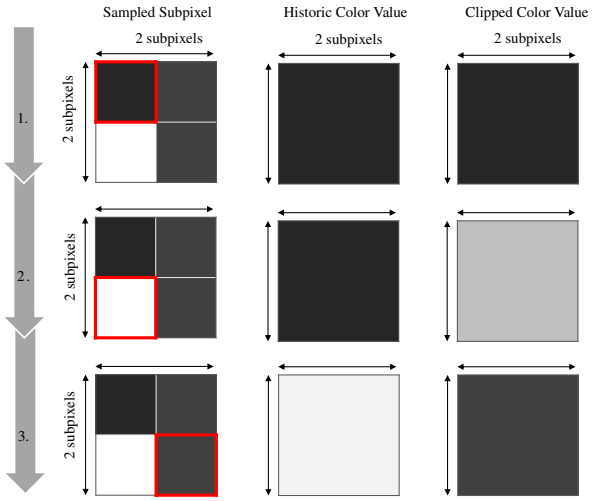


Figure 3: For pixels with non-homogeneous subpixel information, depending on the sampled subpixel, a different color is computed. In the first row, the top-left subpixel, marked red, is sampled. The color is dark, the historic color is dark and, therefore, inside the clipping range. In the second row, a bright subpixel is sampled. The historic color is dark and therefore clipped to the bright color value. In the third row, the sampled subpixel is dark, and the bright historic color value is clipped to the darker value. Whenever a subpixel with a diverging color value is sampled, the effective color changes and the pixel flickers.

3.2 Theory

This flickering emerges from undersampling the color signal. This core understanding of flickering leads to our method for controlling the shading rate, namely STeFAS. The basic idea behind STeFAS is to adapt the shading rate to the amount of color difference of consecutive frames. For higher color differences, when clipping would occur, the shading rate is increased to supersampling shading rates. For smaller color differences, when color is inside the clipping window, the shading rate is decreased utilizing coarse shading rates. In the first case, it is assumed that the signal was undersampled. We therefore need to increase the shading rate. In the second case, it can be assumed that the color values were captured at the right sampling rate and enough information is available from the previous frames. For this reason, we can reduce the shading rate and save performance.

The remaining problem is that due to coarse shading, the TAA clipping region of minimum and maximum values of the neighbourhood is not representative because all pixels inside the coarse pixel get assigned the same color. A solution for this would be to simply increase the neighbourhood for computing the clipping range. This introduces a higher performance cost. Variance sampling, introduced by Patney et al. [4] for foveated rendering, gives a superior solution. By computing the first two raw statistical moments, instead of the minimum and maximum, and deriving the mean and standard deviation better thresholds can be defined that are less impacted by outliers. The clipping factors are then defined by the mean plus/minus the standard deviation. As statistical moments are linear quantities they can be pre-filtered with mipmapping. The current shading rate is used to select the appropriate mipmap level.

With these new, statistically based, clipping factors defining reliable thresholds and a better fitting clipping window, the appropriate shading rates for the next frame can be selected. If the color of the previous frame is outside the clipping window, it is very likely that the region was undersampled or ghosting occured. In both cases we should increase the shading rate. Higher color deviations are linearly assigned higher shading rates. If the color lies inside clipping window we can safely assume that the color was sampled at the correct resolution in the previous frames and the shading rate can be linearly decreased. At the clipping windows border, the standard shading rate of 1x1 is assigned.

4 Shading Rate Stabilization

Shading-rate control mechanisms that base the shading rate on the content of the scene, like the technique proposed by Yang et al. [11], face the problem of temporal shading rate instabilities. The reason for this is that the set shading rate directly influences the content of the scene. For example, pixel tiles of the size 4x4 could get assigned a single color value if coarse shading is applied. This 4x4 pixel region is then presented as a homogeneous region instead of, for example, a small color gradient. These small changes in color could affect the next shading rate and therefore the next shading rate is influenced indirectly

by the previous shading rate. Temporal artefacts like flickering occur.

Solutions for this kind of problem could focus on deriving more stable properties from the scene's content. This can be done by letting the set shading rate influence the computation of the properties. Yang et al. [11], for example, introduces two additional thresholds based on the previous shading rate in their algorithm to define the next shading rate. Our solution takes a more general approach and could be used by any shading-rate control mechanism facing temporal artefacts due to shading-rate instabilities.

Instabilities of the shading rate emerge from a lower sampling of the properties computed to define the shading rate. With this understanding, we can apply a modified version of TAA on the shading-rate image itself to do a supersampling of the shading rate over time. This approach directly stabilizes the shading rate and is not dependent on the properties or the algorithm to compute them.

In contrast to color, shading rates are discrete values and, therefore, averaging of shading rates and then rounding them does not reduce temporal instabilities. Our modification of TAA is based on the assumption that if once a shading rate indicating high details was detected we would undersample these details by reducing the shading rate. For this reason, we take the maximum value of the previous and current shading rate. This effectively avoids undersampling but once a region requiring a high shading rate is observed, this high shading rate would stay at it forever and effectively disable VRS. In a natural scene, after some camera movements, all regions would be shaded with the highest rate. We call this effect shading-rate ghosting.

To resolve it we apply the clipping approach Karis [3] used to recude ghosting to our modified version of TAA too. By clipping the previous shading rate to a window around the current shading rate we can alleviate this kind of shading-rate ghosting. However, the clipping region can not be defined with the neighbouring shading rates. The reason for this is that the regions of shading rates with a size of 16x16 pixels are too large and the same problem as with averaging of the discrete values of the shading-rate would occur. A stable shading rate with little ghosting can be achieved by applying a preference for higher shading rates as the clipping window. As a result, we define our clipping window as the current shading plus two higher rates as the upper bound and minus one lower shading rate as the lower bound.

5 Algorithm

A high-level overview of our STeFAS algorithm including shading-rate stabilization is depicted in Figure 4 (Issued on the CPU) and Figure 5 (Issued on the GPU).

The first frame is rendered with the highest shading rate by initializing the color-difference image to white. This is equivalent to a maximal deviation from the previous frame. Our algorithm starts by clearing the shading-rate image to the highest shading rate to avoid artefacts from previous frames. In the next step, the color-differences image is scaled down to $\frac{1}{16}$ of its resolution in x and y direction to receive an average color difference for each 16x16 tile. Afterwards, a compute shader uses this scaled-down color-differences image to compute the shading-rate image. This shading-rate image is then set up for the VRS pipeline and used during all subsequent draw calls. Finally, TAA is applied to the rendered image. During this post-processing of TAA, the clipping distances are saved in the color-difference image to be used for the next frame.

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 coarse shading rates as described in Section 3. 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 as described by Karis [3] for TAA. These world coordinates are then transformed to the screen-space coordinates of the current frame. These new screen-space coordinates are then used to fetch the previous shading rate and stabilize the computed shading rate as described in Section 4. In the last step, the derived and stabilized 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 clear shading rate, set in the first step of the algorithm.

6 Results

As our technique affects the performance and the quality of the computed frames, our evaluation considers both. We capture the average frame times of animations for various configurations and two test scenes for the performance evaluation. For the quality analysis we compute the SSIM and the PSNR of frames at fixed positions of these animations. Hillesland and Yang [2] used these metrics to evaluate the quality differences for their approach on VRS, texel shading. Vaidyanathan et al. [7] used them for the evaluation of a software implementation of a VRS.

The scenes used during the evaluation are displayed in Figure 6 and Figure 7. The first scene consists of Crytek's Sponza[1] and is often used for the evaluation of rendering techniques. The second scene is custom and consists of an island in the sea. Containing large flat surfaces and high geometry details at the palms to be able to compare flat and complex regions in a single scene. We used physically based shading, introducing high fragment shader loads (about 970 SPIR-V instructions).

[1] created by Frank Meinl, modified by Morgan McGuire in 2014 and by Johannes Unterguggenberger in 2018

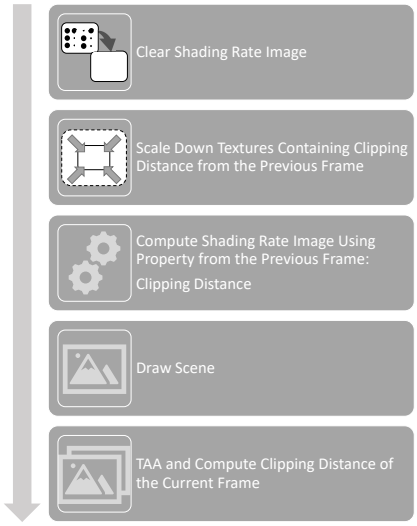


Figure 4: The high level steps of the STeFAS algorithm. These steps are executed inside the render loop in each iteration. In the beginning, the shading-rate image is cleared. Then, the image containing the properties computed during the previous iteration is downscaled and used to compute the shading rate. Subsequently, the scene is drawn using the computed shading-rate image. Finally, the drawn image is anti-aliased using TAA and the clipping distance for the next iteration is computed.

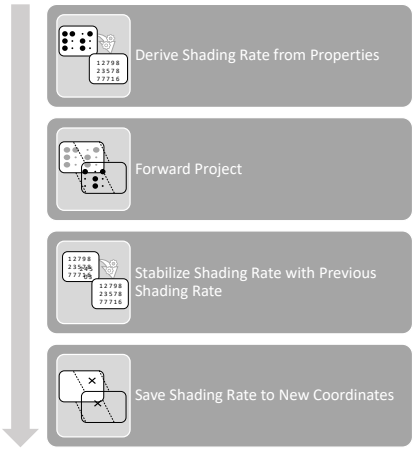


Figure 5: The core steps executed in the compute shader of our STeFAS algorithm. First, the shading rate is derived from the clipping distance. In the next step, the coordinates in the next frame are computed by a forward projection. Finally, the shading rate is stabilized by our TAA-based approach and saved to the new coordinates.

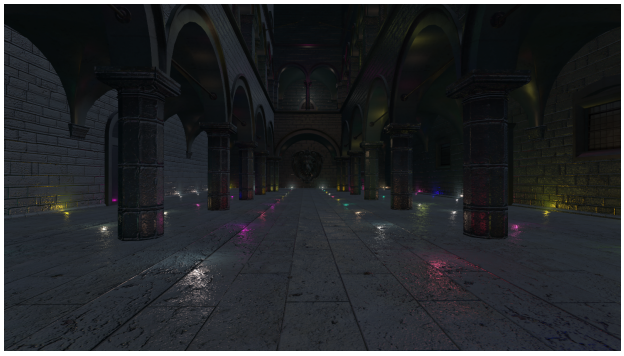


Figure 6: The Sponza scene: Frank Meinl's Crytek Sponza model with detailed normal maps for each surface lit by 100 point lights and one directional light.

To provide a comprehensive comparison we used various sample count and resolution combinations. Our used sample counts reach from 8x MSAA, enabling supersampling shading rates and, therefore, focussing our VRS technique on quality, to more aggressive configurations of 1x MSAA, utilizing only coarse shading rates. Resolutions of 1920x1080, 2560x1440, 3840x2160 and 7680x4320 were used to give an insight how our technique performs for all the currently used and probably in near future usable screen resolutions.

All computations necessary for the evaluation were performed on an Intel Core i7-8700k at a clock speed of 5.0 GHz, 16 GiB DDR4 3200 MHz memory and an RTX 2080 Ti at a clock speed of 1650 MHz featuring 11 GiB GDDR5 1750 MHz memory.

6.1 Quality and Performance

VRS provides supersampling shading rates and coarse shading rates. Depending on the number of MSAA samples configured, different shading rates are enabled. Therefore, we use different sample configurations for the evaluation. Figure 8 shows the performance in frame times and the SSIM in percentage. The ground truth images for the SSIM and PSNR computations were rendered at a resolution of 7680x4320 with 8x MSAA and VRS disabled. Our technique is at least 33% faster in configurations utilizing higher sample rates and focussing on quality. In the performance-wise best cases, utilizing only coarse shading rates, we achieve four times faster frame computations. Alternatively, our technique enables the rendering at higher resolutions, e.g. four times the resolution in x and y coordinates, at the same performance levels of rendering at the lower resolution. For example, rendering with a resolution of 3840x2160 and our VRS technique, we are still 5% faster than rendering at a resolution of 1920x1080 without VRS. This effectively enables higher resolutions while keeping the same frame rates. In terms of quality, the configurations focussing on performance delivery SSIM values with a deviation of 5% - 11%. With the worst case, rendering at a resolution of 1920x1080, SSIM values of 83% are achieved by our technique while images rendered at the same resolution wihtout VRS have a SSIM value of 94%. At higher resolutions and increased sample counts, VRS STeFAS computes images very close to the original with only a single percentage point difference, e.g. at 3840x2160 with 8x MSAA or 7680x4320 with x4

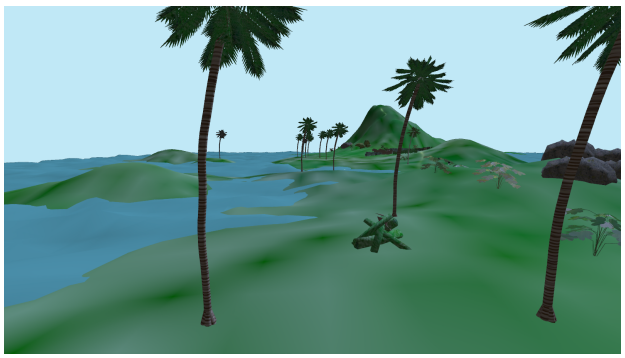


Figure 7: The Island scene: An island in the sea with large flat surfaces and palm leaves consisting of geometry to evaluate geometric aliasing.

MSAA. As expected an increase in samples for our VRS STeFAS technique correlates with an increase in quality. Our evaluation with PSNR values shows similar results as with SSIM.

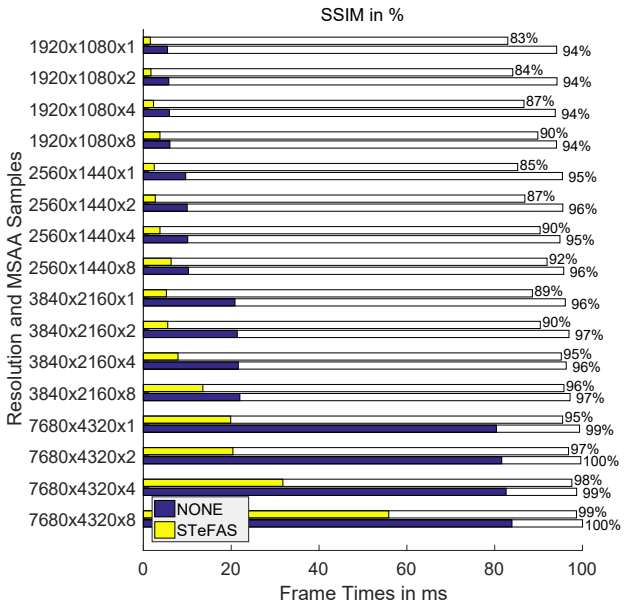
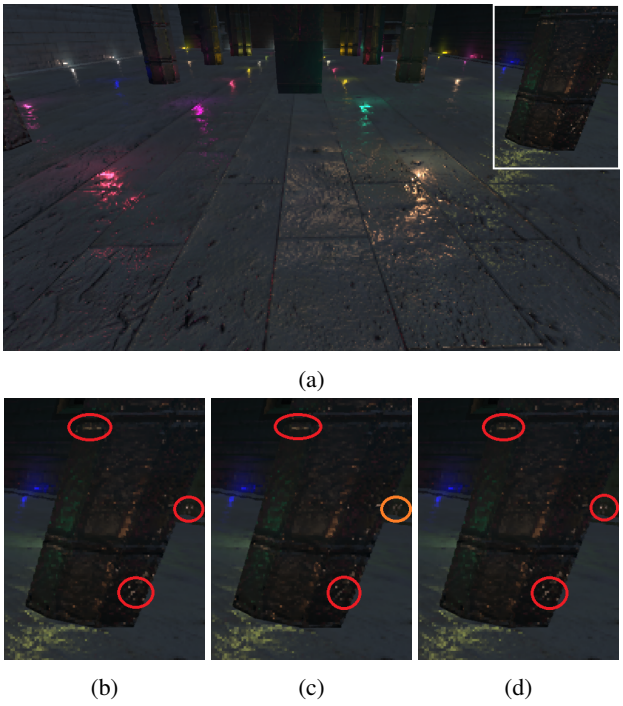


Figure 8: Performance in frame times and quality in SSIM depcited in a single bar chart. The filled color boxes represent the performance. The outlined boxes start at zero as the filled color boxes and represent the SSIM values.

6.2 Temporal Artefacts

In Section 4, shading rate instabilities, the resulting artefacts and our proposed method to reduce them are discussed. While our method and TAA help to alleviate temporal artefacts, they do not vanish completely. To show this effect, we captured three consecutive frames at frame times of 5ms. In Figure 9, one complete frame and excerpts of the three consecutive frames are displayed. Three temporal artefacts are highlighted. We used a resolution of 1600x900 for these images to capture large enough regions that show these temporal artefacts. With higher resolutions or higher MSAA sample configurations, the artefacts are reduced further and at resolutions of 3840x2160 we were not able to detect them anymore. The reason for this might be that these artefacts are aliasing artefacts and at higher sampling rates, the signal is not undersampled anymore even with coarse shading rates.



(a)

(b) (c) (d)

Figure 9: Example of temporal artefacts occurring in the Sponza Scene. The excerpt that is investigated is highlighted in (a) by a white border. In (b),(c) and (d), consecutive frames are displayed and three regions with differences marked by red circles.

7 Conclusion and Future Work

We have proposed two novel techniques, one for controlling the shading rate in order to improving the performance and quality of real-time applications, and one to cope with temporal artefacts that is not only applicable to our shading-rate computation technique but many VRS techniques in general susceptible to these artefacts. Our technique for controlling the shading rate exploits temporal color differences, which, for example, can be derived during TAA, to detected under- and oversampled regions over time and adapt the shading rate accordingly. With our STeFAS approach, we are able to achieve performance improvements of a factor up to four depending on the scene and configuration. These improvements enable higher frame rates, to for example keep the requirements of 90FPS for virtual reality applications, or higher

resolutions to render at native 4k/UHD resolutions while keeping high frame rates for real-time rendering applications. In terms of quality, with VRS STeFAS at the best performance configurations, a deviation of a maximum of 11 percentage points in the SSIM index can be observed. At higher quality configurations, the deviation shrinks to one percentage point at the cost of performance improvements, which shrinks from a factor of four to 33%. Our temporal stabilization algorithm alleviates temporal artefacts due to unstable shading rates and successfully stabilizes the shading rate. It is generally applicable and does not depend on the selection algorithm for the shading rate.

Interesting challenges for the future will be advanced temporal stabilizations of the shading rate and the rendered image. As our eyes are sensitive to change and it captures our focus, changes due to aliasing should be as much avoided as possible. Further improvements in TAA are possible, like increasing the weight for previous frames, and so deliver more temporally stable frames. However, such an approach does not work well with moving objects as ghosting artefacts would introduce even more noticeable change. Another approach could be to find out the correct sampling rate by saving previous color differences and take the lowest shading rate with the smallest color difference to get the best performance while not undersampling the color signal. This would increase the memory consumption and computation time required for the shading rate image.

Another topic for future work can be the combination with other techniques. For example, motion-adaptive shading, adapting the shading rate to the motion of the scene, or foveated rendering, increasing the resolution in the center of the focus of the viewer and decreasing it otherwise. The challenge here lies in how these shading rates should be combined. Simple approaches could focus on performance by taking the minimum of both or focus on quality by taking the maximum of both. More advanced approaches are expected to deliver better results in performance and quality or give a more fine grained control between these two targets. Other combinations could be thought of, as Microsoft has standardized a new version of VRS for DirectX 12 which allows to control the shading rate per primitive, as described by Rhyn, Microsoft [8]. The shading rate defined per primitive is then combined with the shading rate image which allows completely new approaches. For example, higher shading rates could be automatically used for moving objects detected by motion vectors.

In conclusion, VRS STeFAS represents a viable alternative to the few other VRS techniques and a chance for advanced combined approaches. Our algorithm and stabilization technique deliver a workable basis for all kind of VRS techniques.

References

[1] Yong He, Yan Gu, and Kayvon Fatahalian. Extending the graphics pipeline with adaptive, multi-rate shading. *ACM Transactions on Graphics (TOG)*, 33(4):142, 2014.

[2] Karl E Hillesland and JC Yang. Texel shading. In *Proceedings of the 37th Annual Conference of the European Association for Computer Graphics: Short Papers*, pages 73–76. Eurographics Association, 2016.

[3] Brian Karis. High-quality temporal supersampling. *Advances in Real-Time Rendering in Games, SIGGRAPH Courses*, 1:1–55, 2014.

[4] Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke, and Aaron Lefohn. Towards foveated rendering for gaze-tracked virtual reality. *ACM Transactions on Graphics (TOG)*, 35(6):179, 2016.

[5] Jonathan Ragan-Kelley, Jaakko Lehtinen, Jiawen Chen, Michael Doggett, and Frédo Durand. Decoupled sampling for graphics pipelines. *ACM Transactions on Graphics (TOG)*, 30(3):17, 2011.

[6] Ravi Ramamoorthi, Dhruv Mahajan, and Peter Belhumeur. A first-order analysis of lighting, shading, and shadows. *ACM Transactions on Graphics (TOG)*, 26(1):2–es, 2007.

[7] Karthik Vaidyanathan, Marco Salvi, Robert Toth, Tim Foley, Tomas Akenine-Möller, Jim Nilsson, Jacob Munkberg, Jon Hasselgren, Masamichi Sugihara, Petrik Clarberg, et al. Coarse pixel shading. In *Proceedings of High Performance Graphics*, HPG '14, pages 9–18. Eurographics Association, 2014.

[8] Jacques van Rhyn. Variable rate shading: a scalpel in a world of sledgehammers. DirectX Developer Blog, https://devblogs.microsoft.com/directx/variable-rate-shading-a-scalpel-in-a-world-of-sledgehammers/, 2019. online, accessed 09.02.2020.

[9] Kai Xiao, Gabor Liktor, and Karthik Vaidyanathan. Coarse pixel shading with temporal supersampling. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pages 1–7, 2018.

[10] Lei Yang, Diego Nehab, Pedro V. Sander, Pitchaya Sitthi-amorn, Jason Lawrence, and Hugues Hoppe. Amortized supersampling. *ACM Trans. Graph.*, 28(5):135:1–135:12, December 2009.

[11] Lei Yang, Dmitry Zhdan, Emmett Kilgariff, Eric B. Lum, Yubo Zhang, Matthew Johnson, and Henrik Rydgard. Visually lossless content and motion adaptive shading in games. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2(1):6:1–6:18, 2019.