Temporally Adaptive Shading Reuse for Real-Time Rendering and Virtual Reality

JOERG H. MUELLER, THOMAS NEFF, PHILIP VOGLREITER, MARKUS STEINBERGER, and DIETER SCHMALSTIEG, Graz University of Technology

(a) *Sponza* by Crytek (b) *Showdown VR* by Epic Games (c) *Soul City* by Epic Games

Fig. 1. Practical temporally adaptive shading applied to three different scenes using our proposed approach. (a) The *Sponza* test scene contains physically animated boulders and moving spotlights. (b) Our technique implemented in Unreal Engine applied to the *Showdown VR* demo sequence, which contains a large amount of view-dependent effects, including specular highlights and reflections on shiny surfaces. (c) Even large amounts of reflective surfaces as well as animated, wet materials of the *Soul City* scene are handled by our approach without manual fine-tuning. The inlays show the steps underlying our approach. (a) We rely on shading gradients to estimate shading changes and the potential reuse of shading. (b) Using absolute shading differences that are maximum filtered, both per primitive and in an image-space neighborhood, works well in practice. (c) The final shading decision, based on the previously computed shading differences, reduce shading by 57% to 90%, depending on dynamics, while staying visually indistinguishable from full shading in every frame.

Temporal coherence has the potential to enable a huge reduction of shading costs in rendering. Existing techniques focus either only on spatial shading reuse or cannot *adaptively* choose temporal shading frequencies. We find that temporal shading reuse is possible for extended periods of time for a majority of samples, and we show under which circumstances users perceive temporal artifacts. Our analysis implies that we can approximate shading gradients to efficiently determine when and how long shading can be reused. Whereas visibility usually stays temporally coherent from frame to frame for more than 90%, we find that even in heavily animated game scenes with advanced shading, typically more than 50% of shading is also temporally coherent. To exploit this potential, we introduce a temporally adaptive shading framework and apply it to two real-time methods. Its application saves more than 57% of the shader invocations, reducing overall rendering times up to 5× in virtual reality applications without a noticeable loss in visual quality. Overall, our work shows that there is significantly more potential for shading reuse than currently exploited.

CCS Concepts: • **Computing methodologies** → **Rendering**; **Virtual reality**;

Additional Key Words and Phrases: Shading, texture-space shading, virtual reality, temporal coherence, shading difference, temporal shading reuse

Authors' addresses: J. H. Mueller, T. Neff, P. Voglreiter, M. Steinberger, and D. Schmalstieg, Graz University of Technology, Institute of Computer Graphics and Vision, Inffeldgasse 16, Graz, 8010, Styria, Austria; emails: {joerg.mueller, thomas.neff, voglreiter, steinberger, schmalstieg}@icg.tugraz.at.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
0730-0301/2021/02-ART11 $15.00
https://doi.org/10.1145/3446790

ACM Reference format:
Joerg H. Mueller, Thomas Neff, Philip Voglreiter, Markus Steinberger, and Dieter Schmalstieg. 2021. Temporally Adaptive Shading Reuse for Real-Time Rendering and Virtual Reality. *ACM Trans. Graph.* 40, 2, Article 11 (February 2021), 14 pages.
https://doi.org/10.1145/3446790

1 INTRODUCTION

In modern graphics applications, shading typically constitutes a significant part of the workload. Due to the proliferation of complex shading models such as physically based rendering or real-time ray tracing, this trend will certainly continue. Efficient shading is vital for virtual reality (VR) and for energy-efficient mobile graphics.

An important strategy to reduce shading load is to exploit spatial and temporal coherence. In particular, VR rendering demands high frame rates and requires extremely high shading throughput. Recently, increased focus has been put on methods that reduce the spatial resolution of shading. Methods such as checkerboard rendering [El Mansouri 2016], foveated rendering [Patney et al. 2016; Swafford et al. 2016], variable rate shading [Vaidyanathan et al. 2014], and motion-adaptive shading [Yang et al. 2019] enable efficient reuse of spatially coherent shading information.

Temporal coherence is typically exploited for temporal filtering, such as temporal anti-aliasing (TAA) [Karis 2014; Yang et al. 2009], which filters over temporal samples. These techniques aim to implement spatial anti-aliasing with temporal amortization. In contrast, temporal reuse is less frequently exploited, although it presents a significant opportunity [Scherzer et al. 2012]. This is likely owed to the difficulties of predicting how shading points change over time. Even with the aid of advanced caching [Nehab

et al. 2007; Tole et al. 2002; Walter et al. 1999], most methods for exploiting temporal reuse rely on the assumption that shading varies slowly and steadily in the temporal domain. In general, this assumption does not hold beyond a trivial Lambertian model [Herzog et al. 2010]. Therefore, a uniform reduction of the temporal shading rate causes either missed opportunities for savings or leads to visual artifacts.

Although clearly desirable, shading reuse over longer periods of time poses several—at least partially—unresolved research questions:

- Q1 How do temporal artifacts affect the perceived image quality when reusing shading samples over time?
- Q2 What are the limits of shading reuse in scenes with advanced shading and animation?
- Q3 How can we determine ahead of time when shading samples become invalid without actually reshading the samples?
- Q4 Given sufficient temporally coherent shading samples, how can they efficiently be reused in practice?

To close this gap, our work makes the following contributions.

Perception of shading differences. We address Q1 by conducting a controlled user study to determine the perceived effect of shading artifacts due to temporal shading reuse for scenes with advanced shading and animations (Section 3).

Temporal coherence. We address Q2 by analyzing the potential amount of coherence in shading and visibility over time (Section 4). Our experiment reveals that the assumptions on temporal coherence of shading made by previous work on temporal sampling strategies may not hold in many real-world situations.

Gradients. We address Q3 by analyzing analytical and numerical first-order approximations of temporal shading gradients and their ability to predict the magnitude of future shading changes (Section 5). Furthermore, we analyze the spatial variation of the temporal shading gradient and show how to incorporate spatial information to better predict future shading changes.

Framework. We address Q4 with a general-purpose framework for predicting shading changes and temporally reusing shading over time (Section 6). By integrating this framework into two rendering methods, we demonstrate significant performance improvements, particularly for high-throughput VR rendering at an unnoticeable level of quality difference (Figure 1). To evaluate the framework, we conduct two user studies to show the applicability of our method and a detailed analysis of its gains in terms of shading prediction costs, shading cost savings, and overall frame rate gains for a variety of scenes (Section 7).

Overall, our framework reduces the total shading load between 57% and 90%, depending on the dynamics of the rendered sequence. If possible, it retains shading results for multiple seconds without any perceived quality deterioration. These savings make our method the most efficient temporally adaptive shading (TAS) approach that runs on current, unmodified graphics hardware, particularly for VR applications.

2 RELATED WORK

The fact that shading shows spatio-temporal coherence is used in all domains of rendering, ranging from texture mapping to global illumination or denoising. Most related to our approach are techniques that reduce shader invocations by exploiting shading coherence. Techniques such as foveated rendering [Guenter et al. 2012; Kaplanyan et al. 2019; Patney et al. 2016] or variable rate shading [He et al. 2014; NVIDIA 2018; Vaidyanathan et al. 2014] enable efficient reuse of spatial information. Yang et al. [2019] have shown that variable rate shading can result in significant performance gains in modern real-time applications with almost no difference in quality, provided that the system is tuned properly. However, purely spatially adaptive systems lose potential savings in the temporal domain.

Temporal coherence is commonly exploited by using information from previous frames for spatio-temporal filtering. For example, the popular family of TAA techniques [Karis 2014; Yang et al. 2009] uses exponential-decay history buffers for filtering. TAA uses the temporal variation of sampling positions to achieve spatial anti-aliasing. Without any guidance in how quickly shading varies temporally, special care must be taken to avoid ghosting artifacts and low-pass filtering of high-frequency shading changes, for example, using pixel-history linear models [Iglesias-Guitian et al. 2016]. In combination with spatial upsampling, checkerboard rendering [El Mansouri 2016] utilizes a similar approach as TAA for spatio-temporal filtering.

Shading gradients can be used to estimate the variation of shading and are thus often used in spatio-temporal filtering. It is not necessary to use the gradient of the complete shading function, as different shader components can have different frequencies, which can be sampled with different rates [He et al. 2016; Sitthi-Amorn et al. 2008]. Examples for the use of temporal gradients include guiding spatio-temporal upsampling filters [Herzog et al. 2010], denoising filters [Schied et al. 2018], reconstruction in adaptive frameless rendering [Dayal et al. 2005], and spatial sampling [Durand et al. 2005; Ramamoorthi et al. 2007]. Visibility gradients at discontinuities can cause stability problems and require specialized solutions [Li et al. 2018]. Gradients are increasingly used for differential rendering [Kato et al. 2018; Loper and Black 2014] to combine machine learning with graphics. However, they typically focus on gradients of higher-level parameters, such as camera position or material parameters [Liu et al. 2017]. All aforementioned approaches use gradients to guide spatio-temporal filtering or spatial sampling, but they have not been used to guide temporal sampling or shading reuse.

Temporal upsampling methods reuse previous shading results without filtering or accumulation. Asynchronous time warping, popularized by Oculus [Van Waveren 2016], warps the image plane of the previous, fully rendered keyframe based on the latest head-tracking update. Advanced warping and reprojection techniques such as the render cache [Walter et al. 1999] or reverse reprojecetion caching [Nehab et al. 2007] may also use scene depth or motion vectors for dense 3D warping [Didyk et al. 2010; Yang et al. 2011], while reusing the shading from the last keyframe. They are based on the assumption that the temporal variation in shading is slow, and as spatial reprojection errors accumulate over time, a frequent refresh of the cache is required. However, we show that although this assumption holds for a large number of shading samples, a fraction of samples typically violate this assumption and thus lead to perceivable artifacts when not shaded more often.

(a) *Robot Lab* (b) *Sponza* (c) *Space*

Fig. 2. Example views of three test scenes. (a) *Robot Lab* contains animated robots as well as a rotating fan, casting moving shadows. As no lights are animated, *Robot Lab* is our most "calm" test scene. (b) *Sponza* is a recreation of the Atrium Sponza Palace in Dubrovnik modified to include physically animated boulders and animated spotlights, creating fast-moving shadows and overlays of spot lights. (c) *Space* models a large platform in space, including space ships, asteroids, and animated spotlights. It features large color gradients and multiple moving glossy highlights. All scenes use physically based materials.

To avoid spatial reprojection errors while reusing shading samples multiple times, shading can be generated in alternative spaces and resampled for display. A popular example for such an approach is the generation of depth of field and motion blur [Andersson et al. 2014; Ragan-Kelley et al. 2011]. However, for efficiency reasons, shading in alternative spaces often requires GPU extensions [Burns et al. 2010; Clarberg et al. 2014]. Similar in spirit, the shading cache [Tole et al. 2002] has been designed to allow for spatio-temporal shading reuse in path tracing. More recently, texture-space shading methods [Baker 2016; Hillesland and Yang 2016; Tatarinov and Sathe 2018] have been popularized and have even been used for temporal upsampling on a VR client [Mueller et al. 2018]. However, all of these methods only allow for a fixed temporal upsampling rate. In contrast, our method shows how samples can be retained for variable, potentially very long periods of time.

Numerous image quality metrics try to model the human perception of images. Although the peak signal-to-noise ratio (PSNR) provides an objective and easy to compute metric, it fails to capture human perception. The popular structural similarity index measure (SSIM) [Wang et al. 2004], which has been designed to more closely resemble perception, is still the prevalent image quality metric. Various improvements, such as IW-SSIM [Wang and Li 2011], have been made to SSIM to enhance the predictions of the metric. Other metrics, such as HDR-VDP-2 [Mantiuk et al. 2011], even try to model the visual system to some extent. Swafford et al. [2016] build on HDR-VDP-2 to adapt the metric for the evaluation of foveated rendering. Another approach is the combination of different metrics, such as VMAF, which appears especially useful for video game content [Barman et al. 2018]. The novel FLIP [Andersson et al. 2020] metric is derived from the manual method of comparing images by alternating between them and provides an error map showing where differences would be perceived between the two images in comparison. A major disadvantage of all of these metrics is that they only compare images, disregarding any temporal artifacts, such as flickering. A few video quality metrics incorporate temporal aspects [den Branden Lambrecht and Verscheure 1996; Pinson and Wolf 2004; Seshadrinathan and Bovik 2010; Watson 2001; Yang et al. 2007] but focus mostly on video compression–related artifacts such as frame dropping and have not been evaluated for temporal artifacts appearing in rendered imagery. Therefore, conducting a user study remains the gold standard method for the evaluation of perceptual quality, especially when temporal effects have to be considered.

3 PERCEPTION OF SHADING DIFFERENCES

To determine the limits of keeping shading over multiple frames in scenes with advanced shading and animation (and to answer Q1), we conducted a controlled user experiment, in which 34 participants were shown two video clips, one generated with forward rendering as the ground truth reference and the other by reusing shading from previous frames. The participants were asked to rate the relative quality of the two video clips, following a pairwise comparison design [Kiran Adhikarla et al. 2017; Mantiuk et al. 2012]. As the order of clips was randomized, they did not know which clip was the reference. From the rating, we compute an average relative quality score (Q), ranging from −2 to +2, where +2 means the reference is *significantly* better and +1 *slightly* better, and 0 indicates that they have been rated equal. Additionally, we compute the probability p_{ref} of choosing the reference over the reuse approach. A p_{ref} of 50% indicates that there is no difference between the approaches; p_{ref} of 75% is referred to as 1 just-noticeable-difference (JND) unit [Mantiuk et al. 2012]. Staying under 1 JND is considered high quality. For statistical analysis, we use repeated-measures ANOVA and Bonferroni adjustment for post hoc tests. We use the same study design to evaluate various techniques and aspects throughout the article. For all details on the study design, see the supplementary material.

As test scenes, we use three scenes with physically based materials [Schlick 1994], animated models, animated light sources, and dynamic shadows, as shown in Figure 2. *Robot Lab* (Unity) uses the animations coming with the scene, which also include a light shining through a rotating fan that overall has a low number of dynamic changes. *Sponza* (Crytek) was extended with falling boulders and moving, colored spotlights leading to a moderate amount of changes within the scene. *Space*, a scene made from freely available assets from the Unity asset store, contains a metallic platform in space with space ships, asteroids, moving spotlights, and a planet in the background. It is the most dynamic of the scenes, reflecting games with many moving objects and changes on screen at the same time. The lighting in all scenes is based on standard point, spot and directional lights with shadows rendered using shadow mapping. For each scene, we created four 5-second-long test

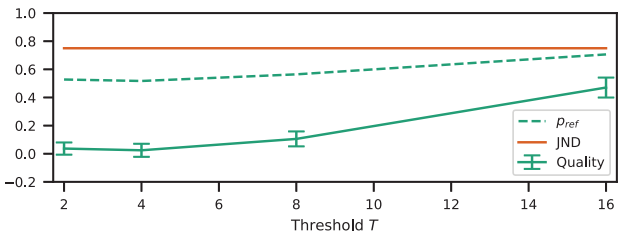


Fig. 3. Participants of our user study rated the quality of rendering with outdated shading using TFR, with a color difference up to a threshold T, in comparison to standard forward rendering. The relative quality score (Q, with a 95% confidence interval) is very close to 0 up to the color difference threshold $T = 8$. p_{ref}, the probability that participants choose forward rendering over TFR, is still close to the balanced 50%. Only at $T = 16$, the probability of detecting a slight difference increases and p_{ref} approaches the JND threshold.

animations, two with a stationary camera and two with a moving camera varying between 0.5 and 6 m/s, which resembles a range of motion from slow walking up to fast running while including fast head rotations. Rendering was restricted to 60 frames per second, the rate of the displays used in the user study. A higher frame rate, such as the 90 frames per second recommended for VR, would lead to even more potential for shading reuse, provided animation frequencies remain unchanged.

To determine the perceived quality reduction caused by reusing shading samples, we implemented an experimental method named *temporal forward rendering* (TFR) that decouples the temporal changes in shading from other major effects that influence the shading reuse ability: temporal changes in visibility and spatial sampling. TFR uses a modified fragment shader to compute shading as if a fragment were shaded at a specific time in the past. We recreate all input parameters to the shader, including view, light and model matrices, textures, and shadow maps for up to 120 frames (2 s) in the past and compare the shading results to the new shading. If the difference of a shading sample (r, g, b) is above a certain threshold T, i.e., $T < \max(|\Delta r|, |\Delta g|, |\Delta b|)$, we consider the shading to be changed. Shading, including gamma-correction and possibly tone mapping, is computed and compared in floating point. However, we use clamped integer values between 0 and 255 for the color components r, g, and b, as well as the threshold T in the following discussion, since these values represent the actual bit depth of the final output. This threshold is an approximation of Weber's law [Blackwell 1972], which states that the just-noticeable luminance difference is constant in relation to the base luminance.

As shown in Figure 3, reusing shading samples that are slightly different does not reduce perceived quality. For $T = 2$ and $T = 4$, Q cannot be separated from 0.0 with confidence, and p_{ref} is nearly 50%. At a threshold of $T = 8$, the mean quality is above 0.0, indicating that some participants see a minor quality deterioration. The distribution is still nearly balanced, with $p_{ref} = 55\%$. At $T = 16$, the distribution is just shy of 1 JND ($p_{ref} = 75\%$). Q is close to 0.5, indicating that, on average, every second participant would rate the difference as "slight." For highest rendering quality, we conclude that temporal artifacts of less than 8 after quantization, i.e., about 3% maximum absolute color difference, are virtually not noticeable by users.

To gain deeper insight, we break down the measurements concerning *Camera* movement and *Scene*. For *Camera*, we found a difference in Q for $T = 16$[1]; the other thresholds are not significant. For *Scene*, we found that there is a difference between *Robot Lab* and *Space* (for $T = 8$) as well as *Robot Lab* and *Sponza* (for $T = 8$ and $T = 16$).[2] Using a larger threshold reveals differences for when shading reuse can be detected. When the camera moves, the image undergoes perspective changes, and it becomes more difficult to see shading imperfections. In contrast, a stationary camera lets the user focus on animated objects, making it easier to notice local shading variations. This assumption was also reassured in our post-study interviews. ("The scenes with moving cameras were more challenging.").

Similar results can also be drawn from *Scene*: *Robot Lab* has the least amount of reflections and abrupt movements. Shadow boundaries move slowly and steadily; there is little overlap between moving objects, making it easy to spot errors. *Sponza* contains a lot of movement, owing to moving spotlights and falling boulders. *Space* is shiny, with large brightness gradients drawing attention.

Combining the results from the in-depth analysis indicates that scenes like *Robot Lab* may have the highest reuse potential, but users may spot quality issues earlier. More dynamic scenes have lower reuse potential, but spotting shading imperfections is also more difficult. Thus, it may be possible to use more aggressive reuse settings in situations where more reshading is required, leading to an overall balanced reuse potential across scenes. One should bear in mind that the study participants were actively looking for differences in shading quality. Distracted observers, such as users engaged in a game, may tolerate even higher thresholds.

4 TEMPORAL COHERENCE FOR SHADING REUSE

Many rendering acceleration techniques exploit temporal reuse but do not provide proof of the method's quality beyond informal comparisons. This leaves the question unanswered to which extent or for how long shading results can be reused in practice (Q2).

4.1 Temporal Coherence of Visibility

To answer the question in detail, we start by recreating the experiment by Nehab et al. [2007] on the degree of temporal coherence found in visibility, applied to our test scenes. We project every sample of a current frame back to the previous frame and determine whether the sample was visible before. In accordance with Nehab et al. [2007], more than 90% of samples stay visible between frames. The most significant visibility disruption is caused by large camera movement or fast-moving objects. For instance, the falling boulders in *Sponza* can lead to visibility changes of up to 50% of the samples.

4.2 Temporal Coherence of Shading

To analyze the potential ideal reuse of shading samples, we rely on the results of our first user study and analyze the number of

[1] Q for $T = 16$: ANOVA $F(1, 33) = 9.506$, $p < .005$.
[2] Q for $T = 8$: $F(2, 66) = 7.348$, $p < .002$, post hoc *Robot Lab* and *Space* ($F(1, 33) = 10.064$, $p < .005$); *Robot Lab* and *Sponza* ($F(1, 33) = 8.1441$, $p < .01$). Q for $T = 16$: $F(2, 66) = 10.499$, $p < .001$, *Robot Lab* and *Sponza* ($F(1, 33) = 24.972$, $p < .001$).

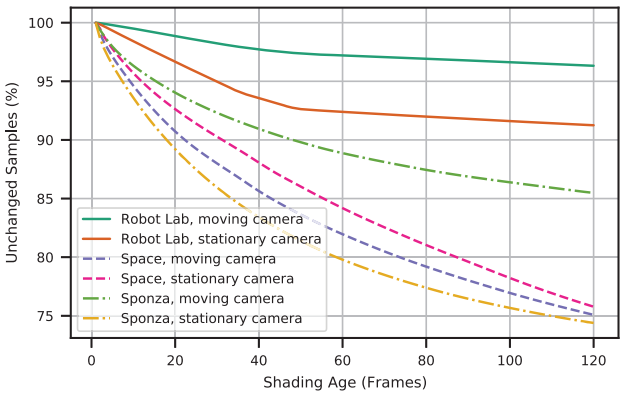


Fig. 4. We use the temporal forward renderer (TFR) to determine how shading behaves independently of changes in visibility and spatial sampling. More than 75% of all samples change less than the color difference $T = 8$ for 120 frames in our test scenes. *Robot Lab* is the least dynamic scene and thus has the least changes. The stationary cameras in this scene's sequences are pointed toward the dynamic changes leading to a lower number of average unchanged samples than for the moving cameras. This is even more true for *Sponza*, where the stationary cameras see more changes than all paths of *Space*. *Space* nevertheless is the most dynamic of the three scenes.

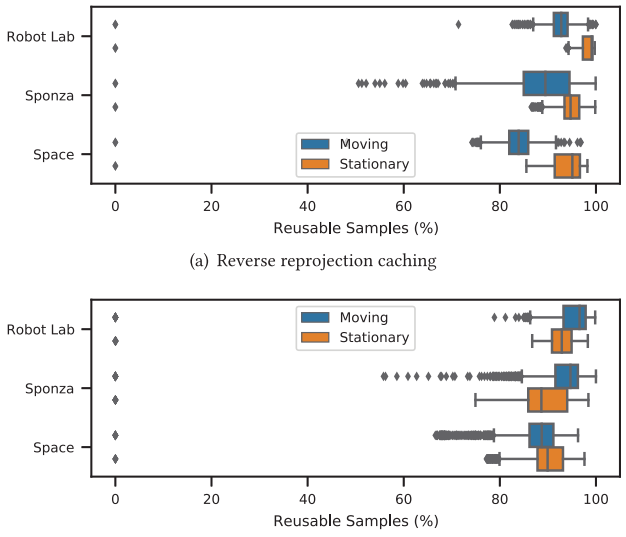


Fig. 5. We determine the possible reuse of two rendering approaches based on visibility and a color difference threshold $T = 8$. (a) Reverse reprojection caching accumulates spatial sampling errors over time, especially when the camera is moving. (b) In contrast, the shading atlas reuse is independent of spatial sampling, and thus the reuse correlates better with the dynamics of the shading as shown in Figure 4. We reuse a block of the shading atlas only if all of the samples within the block can be reused. Outliers at 0 are caused by the first frame of the test sequence in which all samples need to be shaded.

required fragment shader invocation using TFR with a threshold of $T = 8$. Figure 4 shows that in most cases, even for a complex scene such as *Space*, more than 74% of the shading results stay valid (under $T = 8$) for more than 120 frames on average. Being the scene with the least dynamics, *Robot Lab* stays at above 90% over the measured time frame. The stationary cameras in *Sponza* are focused on the falling boulders and thus actually produce more changes of samples than the more complex scene *Space*. A report containing detailed data on other thresholds can be found in the supplementary material.

4.3 Limits of Applying Temporal Coherence

Both the temporal coherence of visibility and the temporal coherence of shading demonstrate a very high potential for reusing shading over many frames. However, practical implementations need to also consider the spatial sampling of shading, i.e., the drift of shading samples, their reprojection error, and the required filtering.

To assess the potential of shading reuse in the light of these issues, we evaluate two practical rendering approaches for shading reuse. The first approach is *reverse reprojection caching* (RRC), proposed by Nehab et al. [2007], which was already used in the experiment for temporal coherence of visibility. RRC reprojects samples from the previous frame to the current frame, potentially accumulating spatial sampling errors. Our implementation runs in two passes, a depth pre-pass and a forward rendering pass that either uses the cache or reshades. To avoid the accumulation of these errors, shading samples can be gathered in a temporally invariant space such as object space or texture space.

Thus, our second approach, *shading atlas* (SA), combines the shading atlas proposed by Mueller et al. [2018] with the rendering pipeline of texel shading [Hillesland and Yang 2016]. In a nutshell, this method shades pairs of triangles in rectangular blocks that are dynamically allocated in a single texture, the shading atlas. The location of the shading samples remains unchanged in the atlas, until the visibility of the triangles changes, or their resolution changes due to a level of detail change, in which case shading is recomputed, i.e., these samples are not reused.

With the two rendering approaches, we evaluate how many samples can be reused from frame to frame. For the RRC, we compute the possible reuse as fraction of the screen resolution, whereas for SA, we consider the fraction of the total allocated space in the atlas in each frame. The results in Figure 5 show that reuse of above 70% could be achieved in most cases. RRC has worse reuse potential with camera movement, as spatial reprojection errors accumulate. SA considers samples reusable only when an entire block is reusable, leading to a slightly worse overall reuse. Less reuse for SA for the stationary cameras in *Robot Lab* and *Sponza* correlates with the temporal coherence of shading changes (see Figure 4), since the camera poses focus on dynamic parts of the scenes.

5 PREDICTING SHADING CHANGES

In the previous section, we established that there is a compelling potential for temporal shading reuse. Several existing methods enable us to map shading samples from one frame to the next either through image-space reprojection or shading in a temporally unaffected space, such as object space or texture space. However, we require efficient prediction of the point in the future when shading samples will become invalid to know how long shading can be

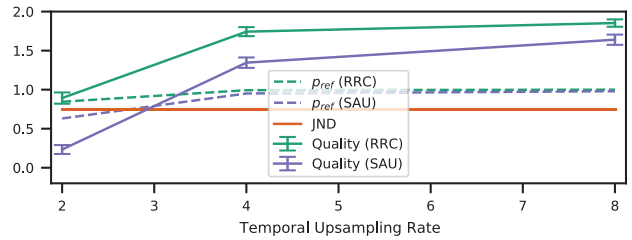


Fig. 6. User study results for RRC and SAU. Whereas SAU is able to maintain a reasonable quality for a 2× upsampling (just below 1 JND), patterns are already visible for RRC with 2× upsampling. Holding shading for longer periods of time is clearly not reasonable.

reused. In this section, we discuss which mechanism is best suited to provide such a prediction.

5.1 Prediction with Fixed Upsampling Rates

Previous strategies rely on uniform temporal upsampling, i.e., shading samples every N^{th} frame. We implemented uniform temporal upsampling in both rendering approaches introduced in Section 4.3, RRC [Nehab et al. 2007] and the shading atlas [Mueller et al. 2018] (SAU). RRC updates 16×16 pixel tiles with a constant refresh rate. A constant fraction of all tiles is updated in each frame, leading to a fixed livespan for each tile. For shading atlas with upsampling (SAU), the livespan of cache entries is also constant, but every cache entry has an individual remaining time to live depending on when it became visible. Both RRC and SAU distribute shading load across multiple frames without noticeable full-screen shading refreshes.

To evaluate how fixed-rate temporal upsampling influences rendering quality, we evaluated RRC and SAU with the same user study design as described in Section 3. As can be seen in Figure 6, uniform upsampling is not able to leverage the potential for shading reuse well, even when reusing shading only once (2× upsampling). RRC at 2× temporal upsampling leads to noticeable differences in 82% of the cases and is, on average, reported to be "slightly worse" in quality. For higher upsampling rates (4 and 8), all participants always noticed differences and reported image quality to be close to "significantly worse." Some of this quality deterioration may be avoidable by using a better filter for temporal upsampling—we used linear interpolation. SAU at an upsampling factor of 2× remains below 1 JND. However, using SAU with upsampling factors larger than 2× is perceived as "significantly worse," with 100% of participants noticing the difference. Overall, we conclude that a uniform upsampling frequency is not sufficient for longer shading reuse.

5.2 Prediction with Shading Gradients

After ruling out fixed-rate upsampling for unlocking the potential of temporal shading reuse, we consider whether mathematical analysis of shading data can be used as a predictor. Arguably, the optimal approach would be a frequency analysis of shading [Durand et al. 2005], but it is also the most expensive (and somewhat unwieldy) option. Ramamoorthi et al. [2007] showed that a first-order gradient analysis is often sufficient in the spatial domain. Therefore, we propose to transfer some of these findings into the temporal domain. We consider a Taylor approximation of a shading function s as an obvious choice for prediction. A simple linear predictor can be formulated as a first-order Taylor expansion from time t_0 to t:

$$s(t) = s(t_0) + s'(t_0) \cdot (t - t_0) + e(t), \tag{1}$$

with a residual error $e(t)$. Based on a color threshold T, we can predict a reshading deadline

$$d = t - t_0 = \frac{T}{s'(t_0)}, \tag{2}$$

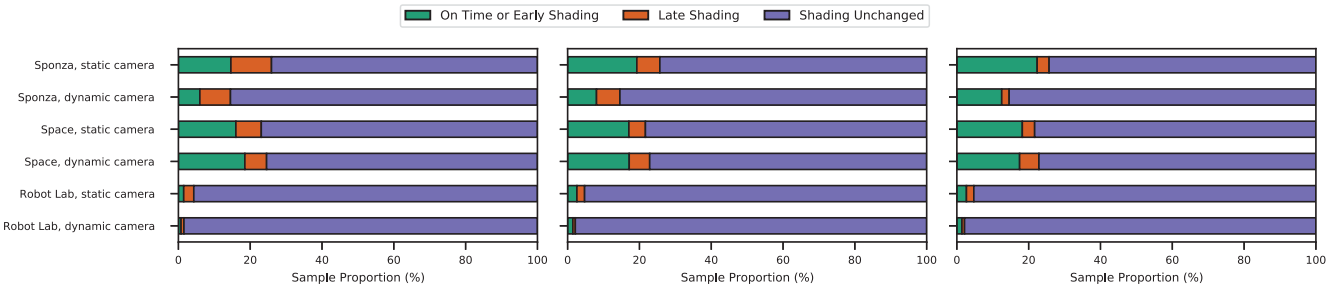
based on the per-channel maximum RGB color gradient $s'(t_0)$.

Analytic derivatives. As a proof of concept, we apply a custom preprocess to automatically differentiate our shader code. Our solution handles not only scalar and vector-valued parameters but also texture lookup (using differences of two texture lookups per spatial dimension) and Poisson-sampled shadow maps. Shader inputs such as camera, object, or light transformations, are extended with their temporal derivatives. For all time-varying parameters where a future state can be calculated deterministically, such as prerecorded animations or physics simulations, we can obtain such derivatives directly by augmenting the animation code. For user-driven inputs, such as camera movement, we can usually compute a gradient based on an extrapolation of the input. Even though the derivatives can be computed alongside the shading, the overhead is non-negligible, resulting in a 1.85× (*Robot Lab*), 1.35× (*Sponza*), and 1.45× (*Space*) average increase of shading runtimes, which would be increased even further by more complex shading models.

Finite differences. An alternative to costly analytic derivatives are finite differences. In the simplest case, we take the backward difference between two shading results. Such backward differences can either be computed between shading in consecutive frames or between frames that are further apart in time. The former has the advantage that it better approximates the limit of finite differences. However, it always requires shading twice in a row. Computing finite differences over a longer interval is more economical and has a potentially beneficial low-pass filtering effect on spurious shading changes. Of course, when a shading sample is first considered, shading must always be done twice. In this way, a first deadline for reshading is extrapolated from the initial gradient. From then on, the long-range gradient is estimated whenever reshading occurs.

Comparison of gradient methods. We evaluate these options for TFR, using the previously found threshold $T = 8$ from the first user experiment. With TFR, we render multiple frames of increasing shading age, resulting in frames with the same sample position but increasingly outdated shading. We use this data to retrospectively obtain the ideal deadline. Starting from the current frame containing the correct shading result, we determine the exact frame in the past where the shading difference exceeds the threshold T. At this point in the past, both the analytical derivatives and finite differences are used to directly predict a future deadline. For the long-range differences, we repeat the process to find the next frame in the past that exceeds the threshold. We limit the search process to 119 frames into the past, effectively clamping the deadline in the range of 1 to 118 frames.

The results of our evaluation are presented in Figure 7 as bar charts separating the samples into ones that are shaded on time



(a) Analytic derivatives (b) Single-frame differences (c) Long-range differences

Fig. 7. We use TFR to evaluate different approaches to compute temporal gradients of shading and how well they predict future changes in shading. Late shading would cause artifacts in the final image output and thus should be avoided. Shading too early may harm performance but does not lower quality. Ideally, a technique avoids late shadings completely while keeping early shadings as low as possible. In accordance with our test for temporal coherence of shading, most samples stay unchanged for more than 120 frames. Note that the simple long-range differences (between two shading points) show the least amount of late shadings while having only slightly increased early shading.

(a) Original (b) Gradients (c) Filtered gradients

Fig. 8. (a) A partial view of *Robot Lab* with moving robots and faint but noticeable shadows slowly moving across the floor. (b) Whereas small shading gradients are typical for smoothly changing shading, such as around the robot or the rotating arm, moving shadow boundaries show steep localized gradients. The two white, diagonal lines represent the shadow borders on the floor. (c) Spatial maximum filtering within 72×72 blocks distributes those gradients to neighbors and correctly predicts where shading should take place. Obviously, such a spatial filter will also predict shading too early, e.g., in the opposite direction of the moving shadow boundary.

or too early, too late, or that do not change at all during 120 frames. Using TFR, this experiment again only considers shading changes and ignores changes in visibility or spatial sampling. As expected from our previous experiments, most of the samples are unchanged for the tested duration, although a few unchanged samples are incorrectly predicted to be in need of shading and thus are shaded too early. Early shading reduces the potential savings.

However, the critical factor for avoiding artifacts is correctly identifying those samples that actually *require* reshading. Recall that a uniform factor of 2× upsampling already significantly reduced perceptual quality. Our experiment places analytic derivatives and single-frame differences very closely, suggesting that finite differences are a good predictor for the analytic derivatives, with long-range differences coming in third with a small penalty. However, all of these methods still miss some required shading, making further analysis necessary.

5.3 Spatial Filtering of Temporal Gradients

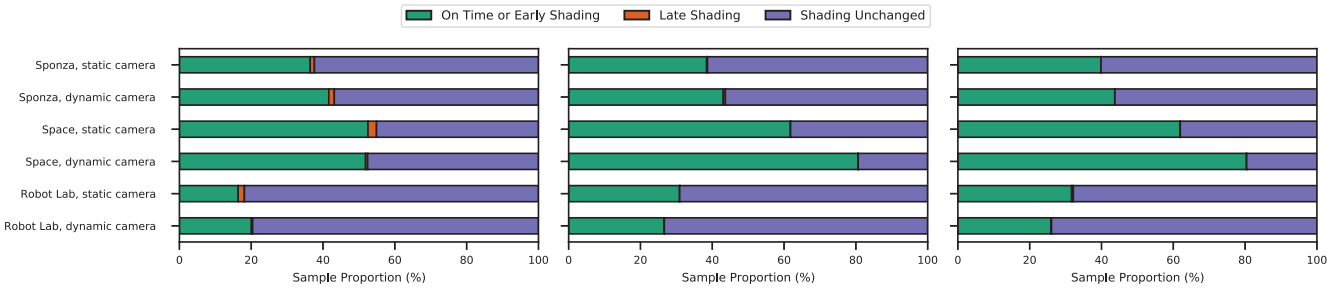
Empirical inspection of the causes for shading gradient mispredictions revealed that the main cause are phenomena that move coherently though space but abruptly change the shading of a single sample. Examples are the boundaries caused by moving lights or shadow borders. This suggests that we must capture the effects of spatially coherent, time-varying changes.

Therefore, we propose a simple maximum filter in image space. This approach is inspired by the render cache [Walter et al. 1999] and shading cache [Tole et al. 2002], which make a shading decision based on the estimated temporal gradient of neighboring samples. The effect of such a filter on shading gradients of a moving shadow border is shown in Figure 8. Clearly, the gradient filtering distributes the highly localized gradients of the shadow boundaries to the surroundings.

We evaluate this filter using TFR to isolate the effect of the spatial filtering while avoiding other sources of misprediction, such as reprojection errors. We use a downsampling factor of 8×8, followed by a convolution with a rectangular kernel size 9×9 (overall touching 72×72 pixels). We empirically determined these parameters to be sufficient, since, at the cost of a lower shading reuse, we avoid late shading overall, which we rate more important to avoid artifacts. When combining the previously described temporal gradient estimation methods with the spatial maximum filtering, we arrive at the results shown in Figure 9. When applying the filter, analytic derivatives, single-frame differences, and long-range differences convincingly avoid shading too late at the cost of more early shadings within the 120 frames. Given the similar properties across these methods, long-range differences are most attractive, since they are the most efficient to compute.

6 TAS FRAMEWORK

Building on our previous findings, we design temporally adaptive shading (TAS) framework, which reliably avoids repeating redundant shading computations, while responding instantly to areas where rapid changes of shading occur. To make the framework largely independent of the rendering algorithm to which it is applied, we introduce the notion of *reuse units* (RUs). An RU is a group of samples for which a uniform decision is made on whether the samples will be shaded anew or shading will be reused. The samples of these units are shaded together, and, consequently, must be stored together in the cache data structure. The renderer determines visibility independently for each unit. For example, a forward or deferred renderer uses a depth buffer to determine per-pixel visibility, whereas other renderer types may determine vis-



(a) Analytic derivatives (b) Single-frame differences (c) Long-range differences

Fig. 9. Applying an image-space filter on top of the gradients strongly reduces late shading compared to Figure 7. Whereas analytic derivatives still lead to late shading, finite differences—in both versions—effectively avoid late shading. Obviously, distributing shading predictions to neighbors increases early shading, leading to less reuse and thus less performance improvement, especially in the highly dynamic scene *Space*. Note that this visualization does not tell us how much reuse is possible, since we cannot distinguish by how many frames shading is early. Please see the supplementary material for the exact distribution of early and late shading.

ibility per primitive. Thus, an RU can be a single pixel as in the case of reverse reprojection caching [Nehab et al. 2007] or a whole block within the shading atlas [Mueller et al. 2018].

With these considerations, we concisely specify the three-step algorithm underlying TAS:

(1) Spatially filtered shading gradients from the last frame are multiplied with the time elapsed since the last shading of each RU and compared to the threshold (T) to decide whether reshading is necessary. Newly visible units are always shaded for two consecutive frames to determine a gradient from finite differences.
(2) The shading is either reused or the unit is reshaded. In the latter case, a new shading difference to the previous shading result is computed for each sample.
(3) The shading gradient is estimated based on the shading difference, scaled by the time difference between them, and a spatial filter is applied to distribute the shading gradient information.

This framework is general enough to be applied to almost any rendering architecture capable of referring to previous shading results.

6.1 Temporally Adaptive Reprojection Caching

In our first reference implementation, the *temporally adaptive reprojection cache* (TARC), extending the method of Nehab et al. [2007], image-space pixels serve as RUs. We replace the periodic refresh of the reverse reprojection caching shader with the first two steps of the framework. Furthermore, we store per-unit and per-sample variables in a double-buffered G-buffer. The input buffers are reprojected, and the potentially altered values are stored in the output buffers. During the depth pre-pass, we also store the estimated shading gradient in the G-buffer. We implement spatial maximum filtering by downsampling the gradient buffer using a maximum filter with overlapping square kernels (as before, we use a downsampling factor of 8×8, followed by a filter of size 9×9). In comparison to standard reverse reprojection caching, TARC needs additional memory for screen size buffers to store the shading difference and the time since the last shading.

6.2 Temporally Adaptive Shading Atlas

Our second reference implementation, the *temporally adaptive shading atlas* (TASA), adds temporally adaptive shading to the SA method. Using a texture-space representation for storing shading samples avoids the accumulation of reprojection errors faced by TARC. It is also convenient to define RUs by proximity of shading samples on object surfaces (or by general proximity in a 3D scene). The RUs in TASA correspond to two triangles packed into a rectangle of $2^N \times 2^M$ texels, where each unit's size in the atlas is determined based on its image-space projection, as proposed by Mueller et al. [2018]. However, other granularities, e.g., 8×8 texels [Hillesland and Yang 2016], per-object texture charts [Baker 2016], or micro-polygons in REYES [Cook et al. 1987], could be chosen.

By retaining the maximum of all shading gradients across an entire RU, a conservative object-space filter is applied to the unit at almost no additional cost, since the samples of an RU are processed together. The resulting object-space filtering is particularly relevant when some of the shading samples are currently occluded in image space.

However, limiting the filter to the boundaries of an RU fails to capture spatial gradients that cross the boundaries of adjacent RUs: For example, a shadow boundary may be creeping slowly across an entire surface consisting of multiple neighboring RUs. We could extend the object-space filter to support a convolution-style kernel larger than a single RU, but this would be a costly operation. Instead, we opt to concatenate the per-RU filter to an image-space filter that determines the maximum over direct image-space neighbors. This image-space filter is not only very inexpensive, but it also captures spatial coherence of perspectively close shading samples, which may not be apparent in object space.

The resulting pipeline works as follows:

(1) Exact visibility is computed per frame in a geometry prepass and stored in a G-buffer as primitive ID with corresponding shading gradients.
(2) Reading the primitive ID, the atlas is updated such that it has room for the visible RUs. The shading gradients are maximum filtered using a 2×2 window in image space for each

RU to propagate the maximum gradient in an image-space neighborhood.

(3) Shading decisions are made on all RUs. RUs for which samples are newly allocated and reallocated in the atlas are always shaded, i.e., they are considered newly visible.
(4) The shading workload is executed including the computation of the shading differences and shading gradients are directly maximum filtered per RU.
(5) The G-buffer is revisited for the final deferred rendering pass.

The additional memory requirements include a copy of the shading atlas to compute the shading differences, per-patch shading differences and times, and a screen space buffer for the spatial filter.

We describe the results obtained with TARC and TASA in the following section.

7 EVALUATION AND RESULTS

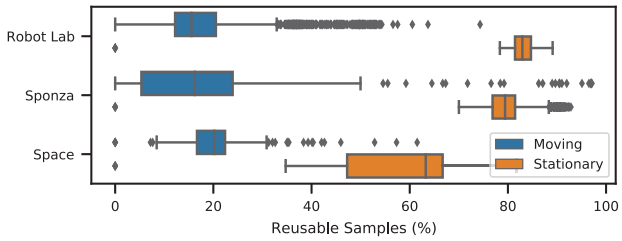
We present results for the reuse, quality, and runtime of our TAS implementations based on technical experiments and two user studies. To avoid sampling artifacts from the atlas when displaying the final image, we test TASA with a 16-MPx atlas (TASA16) and with an 8-MPx atlas (TASA8) to evaluate actual use. Unless stated differently, we use a threshold of $T = 8$ in all experiments, aiming to stay below 1 JND. As the overall goal is reduction of rendering time at high shading loads, we present detailed timing results in comparison to Forward+ rendering. We again use the experimental setup as described in Section 4 with three test scenes and an image resolution of 1920×1080, but we extended the tested sequences to 15 seconds. All tests were run on an Intel Core i7-4820K CPU and an NVIDIA GTX 1080Ti using a custom rendering framework based on Vulkan.

7.1 Reuse

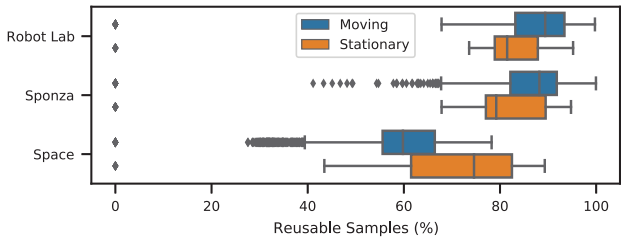
To assess the influence of the TAS algorithm, we evaluate the practical shading reuse for TARC and TASA. In Section 4.3, we discussed the theoretically possible reuse with a perfect prediction of when to shade, resulting in a reuse of 80% to 90% for both TARC and TASA. About 1% to 5% of shading is due to changes in visibility.

As seen in Figure 10, TARC shows low reuse for dynamic camera movements. We found that the reprojection error for camera movements also effects shading gradient predictions, which are slightly too high and, in combination with the spatial filter, invalidate shading often. Although a smaller filter size would increase the reuse potential, it leads to clearly visible artifacts due to missing shading in some scenes. Overall, reuse drops to about 20% for camera movement. For stationary cameras, reuse stays at about 80%, except for *Space*—which shows a lot of motion—where it drops to 50% to 60%. A better reprojection filter for gradients and an adaptively sized image-space filter may increase reuse potential for TARC. However, more advanced filtering and filter size adjustments would also increase overheads.

TASA is able to retain a high amount of reuse (on average 57% to 90%) in comparison to its *ideal* version. The largest drop can again be observed in *Space*, with many high-frequency changes being



(a) Temporally Adaptive Reprojection Caching



(b) Temporally Adaptive Shading Atlas (8 MPx)

Fig. 10. Actual reuse for the TAS implementations with a color difference threshold $T = 8$. TARC has a reasonable reuse for stationary cameras. However, when the camera is moving, the spatial errors introduced by the reprojection, paired with the spatially filtered differences, drop the reuse to impractically low numbers. TASA works significantly better, although it reuses the shading of whole triangle pairs and reshades even when a single sample within the triangle pairs needs reshading. TASA with 16 MPx is nearly identical to 8 MPx and thus omitted here.

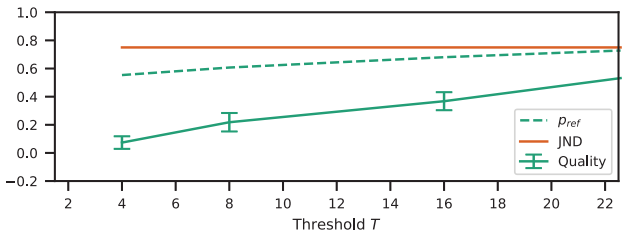
distributed to neighbors. For TASA, the reuse reduction is similar for both stationary and moving cameras, underlining that shading in texture space enables consistent addressing of shading samples. In addition, view-dependent shading effects do not heavily influence shading reuse; only in *Space* with its many highly metallic materials, a moving camera significantly reduces shading reuse.

7.2 Quality

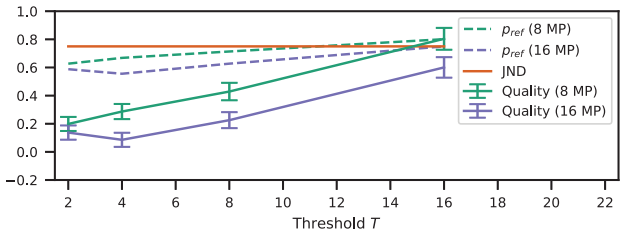
The user study results for both TAS implementations are shown in Figure 11. All videos used in the user study can be found in the supplementary material.[3] For identical target reshading thresholds of T, both approaches behave similarly. For $T = 4$, p_{ref} is close to 50%, and Q is at about 0.1. For $T = 8$, p_{ref} is about 60%, still significantly below 1 JND, and Q is 0.2, indicating a very high quality. A setting of $T = 16$ is about twice as bad in Q and very close to 1 JND, and thus we would suggest to use $T = 8$. For $T = 32$, TARC is already above 1 JND, and Q is close to "slightly worse." We did not find a reason for the slight drop in Q for TASA16 from $T = 2$ to $T = 4$; as the confidence intervals overlap, this may just be a statistical outlier.

A detailed analysis again reveals that there is a significant difference for *Camera* for TARC with $T = 8$ and $T = 16$,[4] as well as TASA16 for $T = 4$ and $T = 16$,[5] again pointing toward the fact that, for dynamic scene content, shading errors are more difficult

[3] Supplementary material can be found online: https://data.icg.tugraz.at/temporally_adaptive_shading/.
[4] Q for $T = 8$: $F(1, 33) = 17.892$, $p < .001$ and $T = 16$: $F(1, 33) = 13.452$, $p < .001$.
[5] Q for $T = 4$: $F(1, 33) = 7.875$, $p < .01$ and $T = 16$: $F(1, 33) = 38.588$, $p < .0001$.



(a) Temporally Adaptive Reprojection Caching



(b) Temporally Adaptive Shading Atlas

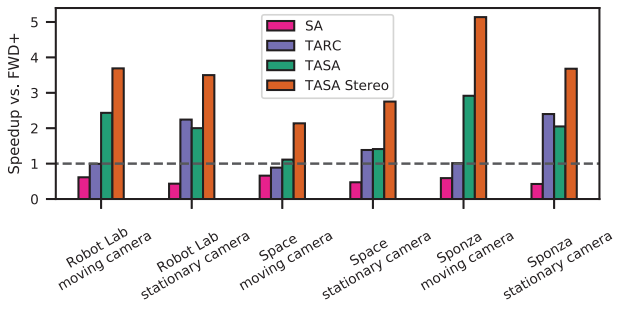
Fig. 11. User study results comparing TARC and TASA with a 16-MPx and 8-MPx atlas. Both approaches show high quality at $T = 8$, staying well below 1 JND. TASA8 loses some quality, as the sample count per surface area is insufficient to generate high-quality images. This effect leads to a constant quality drop in comparison to TASA16, except at $T = 2$, where Q seems to be a statistical outlier for TASA16.

to spot. Similarly, there is a significant difference for *Scene* using TARC with $T = 4$, $T = 8$, and $T = 16$,[6] as well as TASA16 with $T = 8$ and $T = 16$.[7] Post hoc testing revealed that only the difference between *Robot Lab* and the other two scenes was always significant. This fact again confirms the previous consideration that scenes with little movement, but clear moving shading discontinuities, like shadow boundaries, are sensitive with respect to correct shading.

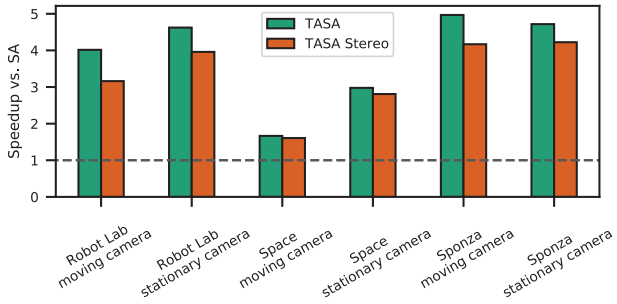
7.3 Runtime

To be of practical value, TAS not only needs to be similar in quality, i.e. rendering at a threshold that is indistinguishable from the ground truth, but must also reduce runtime when including all overheads. Since the approach targets rendering with complex shading, we run our measurements with high shading loads and do not consider any pre- or postprocessing (which are orthogonal to TAS). The overheads of TAS include computing shading differences, spatial filtering, and dynamically deciding whether to shade or not, which may lead to thread divergence during shading and thus reduce the efficiency. To this end, we first measure the overhead of TARC and TASA in addition to the full shading. The overhead is between 14.5% and 16.9% for TARC and between 2.3% and 5% for TASA. This overhead has to be overcome with the shading reuse to perform as well as the rendering approaches without TAS.

The actual speedups are shown in Figure 12. Among the tested scenes, *Space* is especially difficult to speed up using temporal coherence, since most of the scene's surface points are either very



(a) Speedup versus Forward+



(b) Speedup versus SA

Fig. 12. (a) The speedup shows that the overhead of TAS is mostly compensated for TARC, but it fails to accomplish any runtime improvements for moving cameras in contrast to stationary cameras, likely due to the errors caused by repeated reprojection of the shading samples. As expected, SA with an atlas 4× the final output resolution is slower than Forward+. However, TASA manages to compensate the overhead of both the TAS algorithm and the shading atlas, showing considerable runtime improvements for both stationary and moving cameras. When rendering stereo for VR, the benefits of the shading atlas lead to even bigger speedups of TASA in comparison to Forward+ rendering. (b) The spatial filter that needs to be applied twice in stereo rendering causes a slightly higher overhead, leading to a slightly decreased speedup in comparison to SA. Finally, results differ between scenes based on their potential for savings. The highly dynamic space scene performs worst, but still with a considerable speedup.

dynamic or belong to the sky box. The latter can easily be reused but has practically no shading load and thus is unsuitable for TAS. As expected from the limited reuse and the higher overhead, TARC does not improve over Forward+ for moving cameras, but it does have some considerable speedups between 1.38× and 2.4× when the camera is stationary.

Our main focus is on the performance gains of TASA, which is able to re-use shading across the left and right eye buffers in VR stereo rendering. We see that TASA outperforms the other methods for all scenes, both in mono and stereo rendering. The speedup in stereo mode over Forward+ is in the range $2 - 5\times$ ($1.1 - 3\times$ in mono mode). This is noteworthy, as TASA must compensate the overhead of its SA foundation. SA alone is only around half the speed of Forward+ for monoscopic rendering, most likely due to its 8 MPx atlas size that is 4× the resolution of the final output image.

Overall, it can be seen that adaptivity is key for temporal shading reuse. Although uniform temporal reuse strategies reduce shader invocations, quality drops quickly. Using simple shading

[6] Q for $T = 4$: $F(2, 66) = 6.732$, $p < .01$, $T = 8$: $F(2, 66) = 17.129$, $p < .001$, and $T = 16$: $F(2, 66) = 17.394$, $p < .001$.
[7] Q for $T = 8$: $F(2, 66) = 6.366$, $p < .01$, and $T = 16$: $F(2, 66) = 75.338$, $p < .001$.

differences with spatial filtering for gradient estimates works well and is efficient. Especially placing shading samples in texture space appears to be an efficient strategy for reusing them over longer periods of time. However, using an atlas that matches the screen resolution introduces spatial sampling artifacts and reduces sharpness. Interestingly, TAS can easily compensate for these additional shading samples, leading to overall performance gains. However, for techniques that already shade in texture space [Baker 2016; Hillesland and Yang 2016; Mueller et al. 2018] and for VR setups where shading can be reused for both eyes, TAS is highly effective.

7.4 Free-Moving VR Experiment

To demonstrate the flexibility and applicability of our approach for modern high-quality game content, we integrated TASA into *Unreal Engine 4* and conducted a small user experiment in VR. For this purpose, we adapted the *Showdown VR Demo* scene,[8] a slow-motion fly-through of a combat scenario involving several soldiers fighting a giant animated robot. *Showdown* is likely one of the most challenging test scenarios for TAS, as it contains a large amount of highly reflective surfaces, with view-dependent reflections throughout the whole scene, as shown in Figure 1(b). The comparison to the threshold T is evaluated after tone mapping with the Academy Color Encoding System (ACES) Filmic Tonemapper used in Unreal Engine 4.

For our VR user experiment, we slightly modified the scene by subdividing large primitives that exceed the maximum block size in the shading atlas. Additionally, we modified the scene to include fully dynamic directional lighting with cascaded shadow mapping, which was only approximated in the original scene.

Eight participants (six male, and two female, age 24 to 33, with VR experience) tried the SA baseline, followed by TASA configurations using the thresholds $[4, 8, 16, 32, 64]$ and SAU with 4× and 8× upsampling in a randomized order. They were asked to describe their subjective experience during and between all runs. We specifically told all participants to look for visual artifacts of shading, lighting, shadows, and reflections. The atlas size for both SA and TASA was 8 MPx. We used an Intel Core i7-8700K with an NVIDIA RTX 2080Ti and displayed on an HTC Vive at a resolution of 1512×1680 per eye, using a fixed frame rate of 90 Hz.

For TASA with $T = 64$ and $T = 32$, all participants detected artifacts, where $T = 64$ was "very bad" and $T = 32$ was "adequate with some annoying artifacts." For $T = 16$, four participants reported an identical experience compared to the baseline, whereas the remaining four detected minor artifacts on shadows, reflective surfaces, and the soldiers in the scene. These artifacts on the soldiers can be seen in Figure 13 and are related to the muzzle flashes of their guns, where a light source is turned on for a while and then turned off again. Such lighting events are impossible to predict, since large shading gradients (when the lights appear) are interleaved with almost no shading gradient (when the lights stay at constant intensity for several seconds). For $T = 8$, six participants reported an identical experience compared to the baseline, whereas two participants were still able to identify minor artifacts on the soldiers. For $T = 4$, no participant was able to detect any visual artifacts, and all participants reported an identical experience compared to the baseline.

(a) SA baseline (b) TASA ($T = 16$)

Fig. 13. An example of an artifact of the TASA proof of concept running in VR. (a) Ground truth shading of a light fading to black after having a constant intensity for several seconds. (b) The lighting on the helmet is not updating uniformly due to TASA wrongly predicting the shading time, resulting in artifacts at geometric borders.

For SAU with 4× upsampling, four participants reported artifacts related to "jittery" and "flickery" motion, and they reported "low frame rate" for the reflections. When looking at SAU with 8× upsampling, all but one participant reported major artifacts of reflections and shadows, as well as major discomfort, particularly describing the experience as "very uncomfortable when moving around." One participant even reported a mild case of motion sickness. Overall, constant temporal upsampling is more likely to be perceived as jittery, which according to the participants is more discomforting and distracting than the artifacts of TASA, even for large thresholds.

TASA with $T = 8$ resulted in a mostly identical experience to the baseline, in accordance with results in Section 7.2. This configuration shows a reduction of average shader invocations by 65% (Table 1), indicating that TASA is able to effectively reuse shading over long periods of time while shading other regions almost every frame. Due to the different frame rates, results of SAU cannot be directly compared to the results of Section 4, since shading with constant upsampling factors then also runs at different frame rates.

TASA does not depend on slow animations and movement to achieve this result; it handles fast head movements and animations equally well, whereas constant upsampling produces noticeable artifacts. An example of such an artifact can be seen in Figure 14. Avoiding those artifacts is especially important for fast-paced VR gaming, where fast head movements and animations are common.

Overall, our early prototype of TASA in Unreal Engine 4 shows promising results, with no additional fine-tuning required to achieve results that are mostly indistinguishable to the baseline, while reducing shader invocations by 65%, even for the challenging slow-motion scene. Note that to illustrate the capability of our approach to work out of the box, we did not perform any fine-tuning. More scene-specific tuning could increase shading reuse even further.

8 DISCUSSION AND CONCLUSION

Our first objective in this work was to investigate how shading reuse is perceived and could benefit rendering, considering visibility, spatial sampling, and temporal behavior of shading, separated and combined. To this end, we evaluated the perception of

[8] https://www.unrealengine.com/marketplace/en-US/product/showdown-demo

(a) SA baseline (b) SAU, 4× upsampling (c) TASA ($T = 8$)

Fig. 14. (a) A highly view-dependent reflection on the street of the *Showdown VR Demo* scene (bottom left of Figure 1(b)) rendered with the ground truth SA baseline. (b) A constant upsampling rate of 4× produces noticeable artifacts. (c) TASA correctly predicts the shading changes and avoids artifacts independent of camera or animation speed.

Table 1. Average Shader Invocations and Relative Shader Invocation Reductions of the *Showdown VR Demo* Scene Measured for All Tested Methods

Method	Shader Invocations	Reduction to SA
SA	4.49 M	
SAU 4×	1.14 M	74.55%
SAU 8×	0.56 M	87.45%
TASA $T = 4$	2.02 M	55.06%
TASA $T = 8$	1.57 M	65.03%
TASA $T = 16$	0.97 M	78.29%
TASA $T = 32$	0.61 M	86.48%
TASA $T = 64$	0.36 M	91.95%

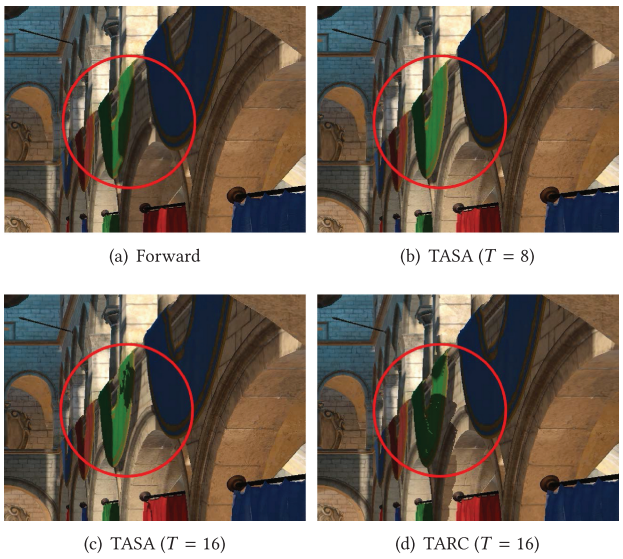
Given roughly equal shader invocations (TASA $T = 16$ vs. SAU 4×, TASA $T = 32$ vs. SAU 8×), study participants prefer the adaptive shading of TASA over the constant upsampling of SAU, showing that TASA provides a more optimal performance-quality tradeoff compared to SAU.

outdated shading using a rendering approach that separates shading from visibility and spatial sampling effects, finding that a shading difference of 3% ($T = 8$ after 8-bit quantization) is not noticed by study participants. We found that even in highly dynamic scenes, many shading samples stay valid for extended periods of time, when considered independently of visibility and spatial sampling.

We found that there is a potential of typically more than 80% shading reuse from frame to frame even in highly dynamic scenes at 60 Hz. The higher frame rates of VR increase this potential further. However, accumulating spatial resampling errors limits the temporal reuse. Thus, we favor texture-space caching of shading samples. Whereas fixed upsampling techniques lead to noticable artifacts even at low upsampling rates, we show that extrapolating shading differences works very well when combined with a simple image-space filter for capturing spatio-temporal effects. Our results demonstrate the benefits of this strategy on shading reuse, image quality, and runtime performance.

8.1 Limitations

The prediction of when to shade is done with a combination of a linear temporal model and a spatial filter. A linear model for temporal changes may be simple and could be improved, but we found it to work sufficiently well. The results of our user study show that the technique is temporally stable, and users hardly noticed any artifacts when a proper threshold is used. Our spatial filter for



(a) Forward (b) TASA ($T = 8$)

(c) TASA ($T = 16$) (d) TARC ($T = 16$)

Fig. 15. TAS cannot predict shading changes that exhibit neither temporal nor spatial coherence. (a) The boulders in our Sponza scene generate distant shadows that first appear outside of the screen and thus are not picked up by our spatial filter. Depending on when shading samples are accidentally reshaded, some of those effects may be picked up: The shadow on the green cloth is completely missed by (b) TASA ($T = 8$); (c) TASA ($T = 16$) and (d) TARC ($T = 16$) capture parts of it. Such failures could be caught with custom improvements, such as projecting shadow maps into the current view.

shading gradients catches effects such as moving highlights and moving shadow boundaries. However, the simple box filter with a rather big kernel size used in TARC leads to considerable amounts of unnecessary shading. A more advanced spatial filter could consider spatial gradients such as optical flow to resolve these issues at the cost of increased runtime and complexity. Although the existing measures capture most changes, some less frequent ones can still cause artifacts—for example, discontinuous rendering, e.g., lights switching on or off, or changes that propagate from outside the image (Figure 15) or from an occluded area. Depending on the use case, specialized cases such as discontinuous changes, e.g. to light sources, can be caught on the scene object level. A more general solution to capture artifacts from discontinuous shading could speculatively update samples that are not due yet.

Furthermore, our evaluation is based on a single threshold applied to the per-channel maximum RGB color difference after tone mapping. This is can be seen as an effective solution based on Weber's law [Blackwell 1972] and similar to the threshold used by Walter et al. [2005]. Our constant threshold assumes a constant base luminance. However, the threshold might be too conservative in certain areas of the HDR spectrum and overlook additional gains. We tested a simple uniform threshold in HDR but did not achieve satisfactory results; a more advanced method may be necessary.

A method for deriving the threshold for noticeable differences from the perception of the human visual system has the potential to lead to further temporal savings. This can possibly be done in a different color space or in the high dynamic range space before tone mapping. For example, a higher threshold could be used

for dark pixels that are close to bright ones, or a lower threshold needs to be used in dark areas where the visual system is more sensitive. The CIEDE2000 [Luo et al. 2001] color difference metric based on the CIELAB color space may prove advantageous over the RGB-based metric and better reflect the perception of luminance [Adelson 2000] and chrominance.

Although object or texture space storage of shading samples has clear advantages over screen space, these methods do come with their own limitations. The shading atlas shades pairs of triangles within rectangular blocks with power-of-2 side lengths. When the level of detail changes, a block of a different size is allocated and shading cannot be reused. The change of the resolution is sometimes noticeable as popping of the textures. Furthermore, the sample distribution within the atlas necessitates some oversampling to achieve the same quality as forward rendering. Although future methods for object or texture space storage for shading reuse may overcome these limitations, the shading atlas in its current form is already practical as a cache for real-time rendering, since it requires no additional GPU extensions and supports streaming.

8.2 Future Work

Temporal reuse and TAS can be applied to other rendering techniques, including global illumination algorithms and ray tracing, where it might be especially useful. Current spatio-temporal filtering techniques for Monte Carlo ray tracing suffer from noisy results in disoccluded areas. TAS would allow to solve this problem by concentrating computational resources in these areas, i.e., immediately computing multiple samples in disoccluded areas, while spending no resources on areas that are predicted to remain unchanged. Additionally, for the overall speedup, it is important to not only consider the non-shading workload, such as the geometry stage, but also pre- and postprocessing, which do not necessarily lend themselves to shading reuse. Motion blur and depth of field benefit greatly from spatial shading reuse, especially in object or texture space as shown by stochastic rasterization literature [Andersson et al. 2014; McGuire et al. 2010]. However, many of the currently used pre- and postprocessing techniques approximate global shading effects, such as shadows and reflections. If, by virtue of shading reuse, more time can be spent on the samples that actually require shading, this benefits the trend for moving global effect computation from postprocessing to ray tracing.

TAS reduces the shading load in areas where no shading change is required, freeing resources for more advanced shading of the remaining scene. Since, in the worst case (discontinuous view change), the whole scene has to be shaded, this can lead to a higher variability in frame rate: Only some frames can be accelerated; others remain at the baseline speed. In our measurements, the absolute frame time variability was unchanged in comparison to the baseline, whereas the mean frame time was reduced. The frame time variability depends mostly on the complexity of the current view, e.g., close-up view vs. wide overview. Lower frame rate variability could be obtained by using TAS as an oracle for a scheduling technique, which uses the predicted shading differences as priority, instead of making decision based on a fixed threshold. This works especially well with our observation in Section 3 that fast-moving scenes make it harder to detect errors in comparison to a still image.

As a technique focused on temporal reuse of shading, TAS is orthogonal to spatially adaptive techniques. Thus, it can be easily combined with spatial reuse of sampling, such as variable rate shading, foveation, and checkerboard rendering. Further research combining these techniques may also reveal further insight into how to optimize the spatial filter that TAS uses. Our hope is that the future will bring more physically correct shading and fewer approximations that require preprocessing steps, like rendering shadow maps, or postprocessing, like screen-space effects in deferred rendering.

Overall, we have shown that the huge potential for temporally adaptive shading reuse can be exploited in modern rendering engines. Doing so may be slightly more difficult than spatially adaptive shading. However, practical performance gains can be achieved even with simple means. We are confident that temporally adaptive shading reuse will become more important as higher-quality shading and real-time global illumination gain traction.

REFERENCES

Edward H. Adelson. 2000. Lightness perception and lightness illusions. In *The New Cognitive Neurosciences*, Michael S. Gazzaniga (Ed.). MIT Press, Cambridge, MA, 339–351.

Magnus Andersson, Jon Hasselgren, Robert Toth, and Tomas Akenine-Möiler. 2014. Adaptive texture space shading for stochastic rendering. *Computer Graphics Forum* 33, 2 (May 2014), 341–350. DOI: https://doi.org/10.1111/cgf.12303

Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D. Fairchild. 2020. FLIP: A difference evaluator for alternating images. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 3, 2 (Aug. 2020), Article 15, 23 pages. DOI: https://doi.org/10.1145/3406183

Dan Baker. 2016. Object Space Lighting. *Game Developers Conference*. Retrieved on Februar 16, 2021 from https://www.gdcvault.com/play/1023511/Advanced-Graphics-Techniques-Tutorial-Day

Nabajeet Barman, Steven Schmidt, Saman Zadtootaghaj, Maria G. Martini, and Sebastian Möller. 2018. An evaluation of video quality assessment metrics for passive gaming video streaming. In *Proceedings of the 23rd Packet Video Workshop (PV'18)*. ACM, New York, NY. DOI: https://doi.org/10.1145/3210424.3210434

H. Richard Blackwell. 1972. Luminance difference thresholds. In *Visual Psychophysics*. Springer, Berlin, Germany, 78–101. DOI: https://doi.org/10.1007/978-3-642-88658-4_4

Christopher A. Burns, Kayvon Fatahalian, and William R. Mark. 2010. A lazy object-space shading architecture with decoupled sampling. In *Proceedings of the Conference on High Performance Graphics (HPG'10)*. 19–28. http://dl.acm.org/citation.cfm?id=1921479.1921484.

Petrik Clarberg, Robert Toth, Jon Hasselgren, Jim Nilsson, and Tomas Akenine-Möller. 2014. AMFS: Adaptive multi-frequency shading for future graphics processors. *ACM Transactions on Graphics* 33, 4 (July 2014), Article 141, 12 pages. DOI: https://doi.org/10.1145/2601097.2601214

Robert L. Cook, Loren Carpenter, and Edwin Catmull. 1987. The Reyes image rendering architecture. *ACM SIGGRAPH Computer Graphics* 21, 4 (Aug. 1987), 95–102. DOI: https://doi.org/10.1145/37402.37414

Abhinav Dayal, Cliff Woolley, Benjamin Watson, and David Luebke. 2005. Adaptive frameless rendering. In *Proceedings of the 16th Eurographics Conference on Rendering Techniques (EGSR'05)*. ACM, New York, NY, 265–275. DOI: https://doi.org/10.1145/1198555.1198763

Christian J. Van den Branden Lambrecht and Olivier Verscheure. 1996. Perceptual quality measure using a spatiotemporal model of the human visual system. In *Digital Video Compression: Algorithms and Technologies 1996*, Vasudev Bhaskaran, Frans Sijstermans, and Sethuraman Panchanathan (Eds.). SPIE. DOI: https://doi.org/10.1117/12.235440

Piotr Didyk, Elmar Eisemann, Tobias Ritschel, Karol Myszkowski, and Hans-Peter Seidel. 2010. Perceptually-motivated real-time temporal upsampling of 3D content for high-refresh-rate displays. *Computer Graphics Forum* 29, 2 (May 2010), 713–722. DOI: https://doi.org/10.1111/j.1467-8659.2009.01641.x

Frédo Durand, Nicolas Holzschuch, Cyril Soler, Eric Chan, and François X. Sillion. 2005. A frequency analysis of light transport. *ACM Transactions on Graphics* 24, 3 (July 2005), 1115–1126. DOI: https://doi.org/10.1145/1073204.1073320

Jalal El Mansouri. 2016. Rendering Rainbox Six Siege. *Game Developers Conference*. Retrieved on February 16, 2021 from https://www.gdcvault.com/play/1022990/Rendering-Rainbow-Six-Siege.

Brian Guenter, Mark Finch, Steven Drucker, Desney Tan, and John Snyder. 2012. Foveated 3D graphics. *ACM Transactions on Graphics* 31, 6 (Nov. 2012), Article 164, 10 pages. DOI: https://doi.org/10.1145/2366145.2366183

Yong He, Tim Foley, and Kayvon Fatahalian. 2016. A system for rapid exploration of shader optimization choices. *ACM Transactions on Graphics* 35, 4 (July 2016), 1–12. DOI: https://doi.org/10.1145/2897824.2925923

Yong He, Yan Gu, and Kayvon Fatahalian. 2014. Extending the graphics pipeline with adaptive, multi-rate shading. *ACM Transactions on Graphics* 33, 4 (July 2014), Article 142, 12 pages. DOI: https://doi.org/10.1145/2601097.2601105

Robert Herzog, Elmar Eisemann, Karol Myszkowski, and Hans-Peter Seidel. 2010. Spatio-temporal upsampling on the GPU. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D'10)*. ACM, New York, NY. http://graphics.tudelft.nl/Publications-new/2010/HEMS10a.

Karl E. Hillesland and J. C. Yang. 2016. Texel shading. In *Proceedings of the 37th Annual Conference of the European Association for Computer Graphics: Short Papers (EG'16)*. 73–76. DOI: https://doi.org/10.2312/egsh.20161018

Jose A. Iglesias-Guitian, Bochang Moon, Charalampos Koniaris, Eric Smolikowski, and Kenny Mitchell. 2016. Pixel history linear models for real-time temporal filtering. *Computer Graphics Forum* 35, 7 (Oct. 2016), 363–372. DOI: https://doi.org/10.1111/cgf.13033

Anton S. Kaplanyan, Anton Sochenov, Thomas Leimkuhler, Mikhail Okunev, Todd Goodall, and Gizem Rufo. 2019. DeepFovea: Neural reconstruction for foveated rendering and video compression using learned statistics of natural videos. *ACM Transactions on Graphics* 38, 6 (Nov. 2019), 1–13. DOI: https://doi.org/10.1145/3355089.3356557

B. Karis. 2014. High Quality Temporal Supersampling. *Advances in Real-Time Rendering in Games course. Presented at the41st International Conference and Exhibition on Computer Graphics and Interactive Techniques (Vancouver SIGGRAPH'14)*.

Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. 2018. Neural 3D mesh renderer. In *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, Los Alamitos, CA. DOI: https://doi.org/10.1109/cvpr.2018.00411

Vamsi Kiran Adhikarla, Marek Vinkler, Denis Sumin, Rafal K. Mantiuk, Karol Myszkowski, Hans-Peter Seidel, and Piotr Didyk. 2017. Towards a quality metric for dense light fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 58–67.

Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. 2018. Differentiable Monte Carlo ray tracing through edge sampling. *ACM Transactions on Graphics* 37, 6 (Dec. 2018), 1–11. DOI: https://doi.org/10.1145/3272127.3275109

Guilin Liu, Duygu Ceylan, Ersin Yumer, Jimei Yang, and Jyh-Ming Lien. 2017. Material editing using a physically based rendering network. In *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV'17)*. IEEE, Los Alamitos, CA. DOI: https://doi.org/10.1109/iccv.2017.248

Matthew M. Loper and Michael J. Black. 2014. OpenDR: An approximate differentiable renderer. In *Computer Vision—ECCV 2014*. Springer International, 154–169. DOI: https://doi.org/10.1007/978-3-319-10584-0_11

M. R. Luo, G. Cui, and B. Rigg. 2001. The development of the CIE 2000 colour-difference formula: CIEDE2000. *Color Research & Application* 26, 5 (2001), 340–350. DOI: https://doi.org/10.1002/col.1049

Rafał Mantiuk, Kil Joong Kim, Allan G. Rempel, and Wolfgang Heidrich. 2011. HDR-VDP-2. *ACM Transactions on Graphics* 30, 4 (July 2011), 1–14. DOI: https://doi.org/10.1145/2010324.1964935

Rafał K. Mantiuk, Anna Tomaszewska, and Radosław Mantiuk. 2012. Comparison of four subjective methods for image quality assessment. *Computer Graphics Forum* 31 (2012), 2478–2491.

M. McGuire, E. Enderton, P. Shirley, and D. Luebke. 2010. Real-time stochastic rasterization on conventional GPU architectures. In *Proceedings of the Conference on High Performance Graphics (HPG'10)*. 173–182. http://dl.acm.org/citation.cfm?id=1921479.1921505.

Joerg H. Mueller, Philip Voglreiter, Mark Dokter, Thomas Neff, Mina Makar, Markus Steinberger, and Dieter Schmalstieg. 2018. Shading atlas streaming. *ACM Transactions on Graphics* 37, 6 (Nov. 2018), Article 199. DOI: https://doi.org/10.1145/3272127.3275087

Diego Nehab, Pedro V. Sander, Jason Lawrence, Natalya Tatarchuk, and John R. Isidoro. 2007. Accelerating real-time shading with reverse reprojection caching. In *Proceedings of the 22nd ACM SIGGRAPH/EUROGRAPHICS Symposium on Graphics Hardware (GH'07)*. 25–35. http://dl.acm.org/citation.cfm?id=1280094.1280098.

NVIDIA. 2018. NVIDIA Turing GPU Architecture Whitepaper. Retrieved January 5, 2020 from https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/technologies/turing-architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf

Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke, and Aaron Lefohn. 2016. Towards foveated rendering for gaze-tracked virtual reality. *ACM Transactions on Graphics* 35, 6 (Nov. 2016), Article 179, 12 pages. DOI: https://doi.org/10.1145/2980179.2980246

M. H. Pinson and S. Wolf. 2004. A new standardized method for objectively measuring video quality. *IEEE Transactions on Broadcasting* 50, 3 (Sept. 2004), 312–322. DOI: https://doi.org/10.1109/tbc.2004.834028

Jonathan Ragan-Kelley, Jaakko Lehtinen, Jiawen Chen, Michael Doggett, and Fredo Durand. 2011. Decoupled sampling for graphics pipelines. *ACM Transactions on Graphics* 30, 3 (May 2011), 1–17. DOI: https://doi.org/10.1145/1966394.1966396

Ravi Ramamoorthi, Dhruv Mahajan, and Peter Belhumeur. 2007. A first-order analysis of lighting, shading, and shadows. *ACM Transactions on Graphics* 26, 1 (Jan. 2007), 2–es. DOI: https://doi.org/10.1145/1189762.1189764

Daniel Scherzer, Lei Yang, Oliver Mattausch, Diego Nehab, Pedro V. Sander, Michael Wimmer, and Elmar Eisemann. 2012. Temporal coherence methods in real-time rendering. *Computer Graphics Forum* 31, 8 (Sept. 2012), 2378–2408. DOI: https://doi.org/10.1111/j.1467-8659.2012.03075.x

Christoph Schied, Christoph Peters, and Carsten Dachsbacher. 2018. Gradient estimation for real-time adaptive temporal filtering. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 1, 2 (Aug. 2018), 1–16. DOI: https://doi.org/10.1145/3233301

Christophe Schlick. 1994. An inexpensive BRDF model for physically-based rendering. *Computer Graphics Forum* 13, 3 (Aug. 1994), 233–246. DOI: https://doi.org/10.1111/1467-8659.1330233

K. Seshadrinathan and A. C. Bovik. 2010. Motion tuned spatio-temporal quality assessment of natural videos. *IEEE Transactions on Image Processing* 19, 2 (Feb. 2010), 335–350. DOI: https://doi.org/10.1109/tip.2009.2034992

Pitchaya Sitthi-Amorn, Jason Lawrence, Lei Yang, Pedro V. Sander, and Diego Nehab. 2008. An improved shading cache for modern GPUs. In *Proceedings of the 23rd ACM SIGGRAPH/EUROGRAPHICS Symposium on Graphics Hardware (GH'08)*. 95–101. http://dl.acm.org/citation.cfm?id=1413957.1413972.

Nicholas T. Swafford, José A. Iglesias-Guitian, Charalampos Koniaris, Bochang Moon, Darren Cosker, and Kenny Mitchell. 2016. User, metric, and computational evaluation of foveated rendering methods. In *Proceedings of the ACM Symposium on Applied Perception (SAP'16)*. ACM, New York, NY. DOI: https://doi.org/10.1145/2931002.2931011

Andrei Tatarinov and Rahul Sathe. 2018. Texture Space Shading. *SIGGRAPH 2018 Exhibitor Talks*. Retrieved January 21, 2021 from https://www.youtube.com/watch?v=Rpy0-q0TyB0.

Parag Tole, Fabio Pellacini, Bruce Walter, and Donald P. Greenberg. 2002. Interactive global illumination in dynamic scenes. *ACM Transactions on Graphics* 21, 3 (July 2002), 537–546. DOI: https://doi.org/10.1145/566654.566613

K. Vaidyanathan, M. Salvi, R. Toth, T. Foley, T. Akenine-Möller, J. Nilsson, J. Munkberg, et al. 2014. Coarse pixel shading. In *Proceedings of High Performance Graphics (HPG'14)*. 9–18. http://dl.acm.org/citation.cfm?id=2980009.2980011.

J. M. P. Van Waveren. 2016. The asynchronous time warp for virtual reality on consumer hardware. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST'16)*. ACM, New York, NY, 37–46. DOI: https://doi.org/10.1145/2993369.2993375

Bruce Walter, George Drettakis, and Steven Parker. 1999. Interactive rendering using the render cache. In *Proceedings of the 10th Eurographics Conference on Rendering (EGWR'99)*. 19–30.

Bruce Walter, Sebastian Fernandez, Adam Arbree, Kavita Bala, Michael Donikian, and Donald P. Greenberg. 2005. Lightcuts: A scalable approach to illumination. *ACM Transactions on Graphics* 24, 3 (July 2005), 1098–1107. DOI: https://doi.org/10.1145/1073204.1073318

Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P. Simoncelli. 2004. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing* 13, 4 (April 2004), 600–612.

Zhou Wang and Qiang Li. 2011. Information content weighting for perceptual image quality assessment. *IEEE Transactions on Image Processing* 20, 5 (May 2011), 1185–1198. DOI: https://doi.org/10.1109/tip.2010.2092435

Andrew B. Watson. 2001. Digital video quality metric based on human vision. *Journal of Electronic Imaging* 10, 1 (Jan. 2001), 20. DOI: https://doi.org/10.1117/1.1329896

Kai-Chieh Yang, C. C. Guest, K. El-Maleh, and P. K. Das. 2007. Perceptual temporal quality metric for compressed video. *IEEE Transactions on Multimedia* 9, 7 (Nov. 2007), 1528–1535. DOI: https://doi.org/10.1109/tmm.2007.906576

Lei Yang, Diego Nehab, Pedro V. Sander, Pitchaya Sitthi-Amorn, Jason Lawrence, and Hugues Hoppe. 2009. Amortized supersampling. *ACM Transactions on Graphics* 28, 5 (Dec. 2009), 1. DOI: https://doi.org/10.1145/1618452.1618481

Lei Yang, Yu-Chiu Tse, Pedro V. Sander, Jason Lawrence, Diego Nehab, Hugues Hoppe, and Clara L. Wilkins. 2011. Image-based bidirectional scene reprojection. *ACM Transactions on Graphics* 30, 6 (Dec. 2011), 1. DOI: https://doi.org/10.1145/2070781.2024184

Lei Yang, Dmitry Zhdan, Emmett Kilgariff, Eric B. Lum, Yubo Zhang, Matthew Johnson, and Henrik Rydgård. 2019. Visually lossless content and motion adaptive shading in games. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 2, 1 (June 2019), Article 6, 19 pages. DOI: https://doi.org/10.1145/3320287

Received August 2020; revised November 2020; accepted January 2021