Coarse Pixel Shading with Temporal Supersampling

Kai Xiao
Intel Corporation

Gabor Liktor
Intel Corporation

Karthik Vaidyanathan
Intel Corporation

Figure 1: The BISTRO scene rendered at 1080p with 2x2 coarse pixel shading, using conventional temporal antialiasing (CPS) and our novel temporal supersampling method (CPS-T). While CPS reduces the shading up to 4x, it also results in a loss of texture detail and shader aliasing (e.g., shadows). Our technique temporally reconstructs shading samples while also preserves the full visibility information. This provides some benefits over 2x checkerboard rendering (CB), which has twice the per-frame shading but only half of the visibility samples. As a reference we show per-pixel shading with TAA (PS).

ABSTRACT
Decoupled sampling techniques such as coarse pixel shading can lower the shading rate while resolving visibility at the full resolution, thereby preserving details along geometric edges. However, while these techniques can significantly reduce shading costs, they also reduce shading quality. In this paper we extend coarse pixel shading with a temporal supersampling scheme that notably improves image quality. We derive multiple shading and visibility samples by jittering each frame using a novel sequence that produces a suitable distribution of samples in both the shading and visibility domain, and temporally resolve these samples to enhance the shading resolution and reduce aliasing. We demonstrate a substantial reduction in shading cost compared to checkerboard rendering, which is another temporal supersampling technique widely used in games today.

CCS CONCEPTS
• Computing methodologies → Antialiasing; Rasterization; Visibility; Graphics processors;

KEYWORDS
rendering, decoupled shading, anti-aliasing

1 INTRODUCTION
Display resolutions have increased significantly in recent years and this trend is likely to continue in the near future, driven by emerging applications such as virtual reality. The corresponding increase in rendering complexity constrained by a limited power budget, presents a major challenge for real time rendering.

Coarse pixel shading (CPS) [He et al. 2014; Vaidyanathan et al. 2014] is proposed to address this problem with a simple modification to the GPU pipeline that allows shading to be sampled at a coarser granularity than a pixel, for example every $2 \times 2$ pixels. Since shading typically accounts for a major portion of rendering cost, this results in a significant improvement in rendering performance. While shading is evaluated at a lower rate, visibility is still computed at full resolution which preserves sharp geometric features and achieves an overall image quality that is better than lower resolution. However, when compared to full resolution rendering, the shading quality can be significantly lower as shown in Figure 1.

Checkerboard rendering (CBR) [de Carpentier 2017; El Mansouri 2016; Wihlidal 2017] is another approach aimed at high resolution displays, where an alternate set of pixels is rendered every frame and two successive frames are resolved to generate a full resolution image. For static scenes this approach can achieve the same quality as full resolution rendering, as samples from the preceding frame...
can perfectly reconstruct the missing samples in the current frame. However with dynamic scenes, samples from the preceding frame may not contribute to the current frame, for example when a sample gets dis-occluded in the current frame. In these scenarios, the missing pixels in the current frame have to be extrapolated from its neighbors which can lead to artifacts as shown in Figure 1. Since checkerboard rendering only lowers the number of samples by half, these artifacts are usually not severe. With coarse pixel shading on the other hand, the shading rate can be lowered independent of visibility, which can enable more aggressive shading reduction.

In this paper we propose a temporal supersampling technique for coarse pixel shading (CPS-T) that shades at a granularity of 2 × 2 pixels and achieves a shading quality that is comparable to checkerboard rendering at a significantly lower cost. Along the lines of previous approaches such as temporal antialiasing [Jimenez et al. 2012; Karis 2014; Yang et al. 2009], we jitter each frame to generate multiple samples. However we show that applying a low discrepancy jitter sequence to the visibility samples can result in a poor distribution of shading samples, and vice versa. To address this problem we extend the void and cluster method [Ulichney 1993] to compute a jitter sequence that generates samples with blue noise characteristics in shading and visibility domain.

We also derive a technique to temporally resolve the jittered frames that takes into account the positions of the shading and visibility samples to provide a better shading resolution in addition to antialiasing. Temporally jittering and accumulating frames can result in flickering and ghosting artifacts which are exacerbated by the coarse granularity of shading samples and a negative LoD bias that we introduce to preserve texture details. Therefore previous techniques to suppress these artifacts based on color clamping [Karis 2014; Patney et al. 2016; Salvi 2016] can be inadequate, especially in scenes with high texture detail. We improve the variance sampling approach by Patney et al. [2016] and adapt the variance sampling interval to varying density of shading samples around geometric edges. Compared to a fixed variance sampling interval this results in significantly fewer artifacts.

Our primary contribution is a temporal supersampling scheme for coarse pixel shading that introduces the following ideas:

- A technique to resolve shading as well as visibility samples across jittered frames
- A color clamping scheme based on adaptive sampling of the color variance to reduce ghosting and flickering artifacts
- A novel jitter sequence that generates a good distribution of shading as well as visibility samples

2 RELATED WORK

Decoupled sampling refers to a wide range of techniques that sample shading at a different rate or in a different domain as compared to visibility. Multisample antialiasing (MSAA) [Akeley 1993] is one of the earliest of such techniques that is widely used today. MSAA samples visibility at several sub-pixel locations but evaluates shading at a coarser granularity of one shading sample per pixel, achieving better quality antialiasing at a reasonable shading cost. As shading complexity and display density have increased in recent years, several other decoupled sampling approaches have been proposed to further reduce shading costs. Coarse Pixel Shading (CPS) [He et al. 2014; Vaidyanathan et al. 2014] extended the shading granularity to several pixels, further reducing the shading rate compared to MSAA. Patney et al. [2016] applied CPS to foveated rendering [Guenter 2007] and aggressively reduced shading based on a perceptual model. While CPS produces acceptable results for specific applications such as foveated rendering or in scenarios where shading lacks high frequency detail, it can lead to poor image quality on mainstream displays.

Ragan-Kelly et al. [2011] applied decoupled sampling to 5D stochastic rasterization by shading in a 2D screen space slice of the 5D sampling domain, achieving a significant reduction in shading cost. Clarberg et al. [2011] extended this approach to deferred rendering where shading was evaluated after visibility was fully resolved, further reducing shading costs. While these techniques lower the shading rate, they still sample shading in screen space which can lead to over-shading in scenes with high geometric complexity as there may be several visible triangles inside a pixel, requiring multiple shader evaluations. Object space shading [Burns et al. 2010; Clarberg et al. 2014; Cook et al. 1987; Hillesland and Yang 2016] can overcome this limitation by sampling shading in object space which enables shading reuse across triangle boundaries. However these systems require a uniquely parameterized domain for shading which can constrain authoring of assets.

Recently, another class of approaches based on temporally amortized supersampling has gained significant popularity. Yang et al. [2009] applied this approach to temporal antialiasing (TAA) and several improvements have been introduced since [Corso et al. 2017; Jimenez 2016; Jimenez et al. 2012; Karis 2014; Salvi 2016]. Beside of antialiasing, temporally amortized sampling can also be used for resolution enhancement. Checkerboard Rendering (CBR) [de Carpentier 2017; El Mansouri 2016; Wihldal 2017] is such a technique that renders half the number of pixels every frame and resolves successive frames to generate an image at the native resolution. However CBR subsamples both shading and visibility by 2× in each frame which can lead to artifacts resulting from missing details in regions with complex visibility. CPS on the other hand samples visibility at the native resolution preserving these details. Our CPS-T technique leverages CPS and temporally amortized supersampling to achieve a greater reduction in shading than checkerboard rendering with a comparable image quality.

3 ALGORITHM

Our algorithm builds upon Temporal Anti-aliasing (TAA) [Jimenez et al. 2012; Karis 2014; Yang et al. 2009] which is an approximate supersampling technique that amortizes the cost of sampling over multiple frames. We start with a description of supersampling and TAA followed by our algorithm.

Let \( p \) represent the 2D pixel coordinates and \( v \), a sample offset. The supersampled color value at a pixel is then given by:

\[
\hat{c}(p) = \sum_{v_i \in \Omega} c(p + v_i)w(v_i),
\]

where \( w \) is a filter function, \( c \) is the sampled color and \( \Omega \) is the set of sample offsets in the filter footprint which can be derived from a low-discrepancy sequence [Halton 1964]. We assume a box filter, where all samples are weighted equally i.e. \( w = \frac{1}{|\Omega|} \).
Coarse Pixel Shading with Temporal Supersampling

Figure 2: CPS2x2 (a) has one shading sample (red) for a group of four adjacent visibility samples (blue). Applying a sub-pixel jitter (b) based on a (2,3) Halton sequence results in a poor sample distribution. Scaling the jitter by two (c) results in a better distribution of shading samples but a poor distribution of visibility samples. Our blue noise jitter sequence results in good shading and visibility distribution.

![Figure 2](image2.png)

Figure 3: A set of decoupled shading (red) and visibility (blue) samples. Each visibility sample corresponds to a visible surface for which shading is sampled and filtered with a shading filter to reconstruct a color value. The reconstructed colors are then filtered with a separate pixel filter.

![Figure 3](image3.png)

TAA approximates this filter with an exponential smoothing function and recursively resolves a single sample every frame. The resolved color at pixel \( p \) and frame \( k \) is given by:

\[
\hat{c}(p) = \alpha w^\beta(u, p) + (1 - \alpha) \hat{c}(p) + \alpha \hat{c}(p),
\]

where \( \alpha \) is a blending factor that controls the exponential falloff. \( \alpha \) can be set as a constant (typically \( \alpha = 0.1 \)), or a dynamic factor adaptive to disocclusion detection. We refer to the resolved color in the previous frame \( \hat{c}(p) \) as the history color.

With decoupled sampling, the shading sample positions can be different from the visibility samples. Assuming \( v \) is the visibility sample offset and \( v^d \) is the shading sample offset, the resolved pixel color is given by:

\[
\hat{c}(p) = \sum_{v_i \in \Omega^v} w^\beta(v_i) \sum_{v_j \in \Omega^\alpha} \hat{c}(p + v_i) w^\alpha(u, p + v_j),
\]

where \( \hat{c}(p + v_i) \) represents the shaded color from a surface that is visible at the visibility sample \( p + v_i \). This shaded color is sampled and filtered with a shading filter \( w^\beta \) to reconstruct a color at the pixel position \( p \). The reconstructed color values corresponding to the visibility samples are then filtered with a pixel filter \( w^\alpha \) as shown in Fig. 3.

Figure 4: A set of shading and visibility samples generated using the void and cluster method (a). We optimize shading sample locations by summing up the shading and visibility weights (b) resulting in a blue noise spectral distribution for both shading (c) and visibility (d) samples.

Along the lines of TAA we process a single sample per frame and approximate Equation 2 with an exponential smoothing function:

\[
\hat{c}(p) = \alpha w^\beta(v) \hat{c}(p + v) + (1 - \alpha) \hat{c}(p).
\]

Here, the pixel filter \( w^\beta \) in decoupled sampling is approximated by the blending factor \( \alpha \), as illustrated in Equation 1. Substituting \( \alpha = \alpha w^\beta(v) \), we get a resolving function similar to TAA but with an exponential falloff factor that is scaled by the shading filter:

\[
\hat{c}(p) = \alpha^w(v) \hat{c}(p + v) + (1 - \alpha^w(v)) \hat{c}(p).
\]

Equation 3 also applies to coarse pixel shading (CPS) which is a decoupled sampling technique. We assume a coarse pixel size of \( 2 \times 2 \) pixels (CPS2x2) where the shading sample is always at the center of four neighboring visibility samples as shown in Fig. 2.

3.1 Blue Noise Jitter Sequence

Sampled color \( c \) can be generated by jittering the viewport for each frame with an offset equal to the sample offset \( v_i \). In CPS, the offsets are sub-pixel values, typically derived from a low discrepancy sequence such as a (2, 3) Halton sequence [Halton 1964].

However with CPS2x2, applying such a jitter sequence results in a trade off between the quality of shading and visibility samples as shown in Fig. 2. In order to derive a jitter sequence that produces a good distribution of shading as well as visibility samples, we leverage the void and cluster method [Ulichney 1993] from the field of digital halftoning. We start with a description of the process for generating sub-pixel jitters and then extend it to CPS.

Let \( g \) be a \( N \times N \) binary mask where \( g(x, y) = 1 \) represents a sample at the cell \( (x, y) \). The mask is initialized with \( N \) samples drawn from a uniform distribution and circularly convolved with a 2D Gaussian function to generate a weight \( h \) for each cell. Circular convolution ensures that the samples can be perfectly tiled. A large value of \( h \) indicates a region where samples are more densely distributed i.e. a cluster and a low value indicates a region where samples are sparse, a void. The sample distribution is iteratively optimized where, at each iteration, a sample is moved from the tightest cluster \( (h_{max}) \) to the largest void \( (h_{min}) \) and \( h \) is re-evaluated. The algorithm stops when the value of \( h_{max} \) does not decrease anymore.

To extend this approach to CPS2x2 we use a mask \( g^d \) for the shading samples and another mask \( g^v \) for the visibility samples. \( g^d \) is derived from \( g^v \) by mapping each shading sample to four visibility samples that wrap around the boundaries of the \( N \times N \) grid. We find that aligning one visibility sample to the shading point produces...
better results as there are fewer unique points to optimize. \(q^*\) is initialized with \(N_0\) random samples and then iteratively optimized. At each iteration, \(q^*\) is generated and then both masks are filtered and summed to identify the void and cluster locations. Clusters that overlap with a shading sample are moved to voids. In this way we optimize the shading sample locations also accounting for the distribution of visibility samples. During rendering, a full set of blue noise offset sequence is pre-computed and used to control the viewport jittering every frame, same as the usage of pre-computed Halton sequence in TAA.

### 3.2 Dynamic Scenes

In dynamic scenes, a geometric surface overlapping a pixel in frame \(k\) might be at a different position in frame \(k-1\). Therefore directly resolving the color value using Equation 1 can result in significant blurring or ghosting artifacts. In order to compensate for this motion, TAA approaches typically sample the screen space velocity for the front most surface around the pixel and its 4-connected neighborhood and then evaluate the corresponding position of the history color in the previous frame.

While this can avoid excessive blurring, ghosting artifacts can still be caused by incorrect velocities resulting from disocclusion i.e. a previously occluded surface becoming visible in current frame.

A simple method to reduce ghosting is to clip the history color against the bounding box of colors in a pixel’s neighborhood [Karis 2014]. A more flexible approach is to evaluate the color variance \(\sigma^2\) over a small interval in the neighborhood of the pixel and clamp the history color to standard deviations \(n\sigma\) from the pixel color [Patney et al. 2016; Salvi 2016]. A larger value of \(n\) trades off flickering for ghosting, and vice versa. Patney et al. [2016] show that sampling the variance at a larger scale can better adapt to the sparse distribution of coarse shading samples. They use mipmaps of color moments to efficiently evaluate the variance at different scales.

**Dynamic Variance Sampling:** Although increasing the variance scale for CPSx2 effectively reduces flickering, it can also lead to additional ghosting. We improve upon this by adaptively lowering the variance scale around geometric edges where a larger number of shading samples are available from overlapping surfaces. We detect such regions by evaluating the absolute difference in velocity between a pixel and the front most surface amongst eight points given by \((x \pm 2, y \pm 2), (x, y \pm 2)\) and \((x \pm 2, y)\), where \((x, y)\) are the coordinates of the pixel. When this difference exceeds a certain threshold (typically, 30% of the maximum velocity in the sampled pixels), we sample variance at the base scale (LoD = 0) and otherwise at a higher scale (LoD = 2). Moreover we also switch to a lower LoD following a dis-occlusion event in order to reduce ghosting.

### 4 IMPLEMENTATION

**CPS Emulation:** Since CPS is not natively supported by any GPU, we emulate CPSx2 by rendering the scene at a quarter resolution with 4X MSAA and then copying the MSAA samples to a full resolution image. We also use the Programmable Sample Positions feature in Direct3D 12, to align the MSSA sample positions with the CPSx2 pixel centers. Both CPS and MSAA compute derivatives with respect to the sparsely distributed shading points, resulting in blurred texture samples. In order to preserve texture detail with our temporal supersampling technique, we negatively bias the LoD by one level when sampling textures. With CPSx2x, alpha tested visibility is resolved at the native resolution. This is achieved by applying alpha testing at visibility sample rate in a depth-only pre-pass.

**Temporal Supersampling:** As shown in Equation 3, our supersampling technique blends a pixel in the current frame with its history color using a blending factor \(\alpha^2 = \alpha w^2(u_j)\), that includes a shading filter \(w^2\). In our implementation, we use a Gaussian filter \(N(\sigma^2)\) with \(\sigma = 0.7\) for this purpose. Also as discussed in Section 3.2, we sample the color variance at a scale that is determined dynamically based on velocity variations around an edge or following a dis-occlusion event. For detecting dis-occlusions, we leverage a technique that is used in some TAA implementations, where the minimum depth in a 3x3 pixel neighborhood of the current frame is compared with the maximum depth in a corresponding 2x2 pixel region in the previous frame. A dis-occlusion event is detected when the current depth exceeds the previous depth.

### 5 RESULTS

In order to evaluate the quality and performance of our method, we have integrated it into a simple game-like rendering framework, derived from the Microsoft MiniEngine D3D12 demo. The main rendering pass of the application implements the Forward+ lighting algorithm, with a single shadow-mapped directional light and several point and cone light sources that are sorted into screen space tiles. By controlling the number of overlapping light sources per pixel, we can provide a pixel shading workload of a realistic complexity. The engine also features some post-processing effects such as tone-mapping, bloom and luminance adaptation. However, we do not use motion blur or depth of field.

We evaluate three scenes with different levels of texture details and geometric complexity. The SPONZA scene has moderate detail with a few high-frequency components such as normal maps and low geometric complexity. Therefore we can expect CPS to effectively reduce the number of pixel shader invocations for this scene. The SANMIGUEL scene has complex geometry with very small triangles and vegetation that uses alpha-masked textures. Finally, the BISTRO scene is the closest one to a modern game workload with high-resolution geometry, textures, and specular surfaces.

For each scene, we compare our method (CPS-T) against the reference full-resolution pixel shading (PS), standard coarse rate shading (CPS), and checkerboard rendering (CBR). All the other methods use the temporal antialiasing implementation in MiniEngine, which we have extended with variance clipping [Salvi 2016]. Our implementation of checkerboard rendering is based on [El Mansouri 2016], which renders half the number of pixels every frame by using a quarter resolution render target with 2X MSAA and per-sample shading. Recent versions of CBR [de Carpentier 2017; Wihlidal 2017] also leverage EQAA but we do not use this feature, since it is only available on specific GPUs and platforms.

#### 5.1 Image Quality

For analyzing quality with all scenes, we captured images from frame sequences that featured a steady, slow-moving camera animation. All sequences were rendered at a constant rate of 60Hz, at
a resolution of 1080p. Fig. 1 shows our comparative results for the BISTRO scene, and Fig. 8 for SPONZA and SANMIGUEL. All methods except standard pixel shading (PS) use reduced-rate sampling of shading or visibility, therefore we use PS as the ground truth for our quality comparisons. Since CPS reduces the shading rate by a factor of 4, it yields images with significant loss in texture detail, due to the LoD bias needed to avoid severe flickering even with TAA. Additionally, it is prone to undersampling artifacts with shading components that are not prefiltered, such as hard shadows. For example, in the SPONZA scene this can be noticed on the curtains. The prefiltering of the normal and specular maps significantly alters the appearance of the lion figure. In the SANMIGUEL insets we can also see a loss of shadow detail.

CPS-T allows the use of a texture LoD that matches per-pixel shading and our reconstruction filter is capable of increasing the perceived shading resolution. However, we can still observe some artifacts compared to PS when either temporal reprojection or color clipping fails to accumulate sufficient samples. For example, we can observe some texture undersampling (Moire artifact) in red curtain of SPONZA, and the loss of detail in foliage shadows in SANMIGUEL.

Note that CBR utilizes twice the amount of shading samples per frame compared to our method. Therefore it is able to match the quality of PS in just two frames, as long as there is perfect correspondence between pixels of alternating frame pairs. When that correspondence fails due to disocclusion or sudden change in shading, CBR is forced to fill in the holes from neighboring pixels, which results in artifacts along edges. We can observe such artifacts in Fig. 1 (top right), the foliage in SPONZA and the thin geometry of the chairs in SANMIGUEL. In contrast, a key benefit of CPS-based techniques is that visibility is always sampled at full resolution, which avoids holes in geometry even under complex animation.

Fig. 5 shows results with our silhouette-aware adaptive color clipping that achieves a balance between flickering and ghosting. Increasing the variance scale introduces some false positives in the disoccluded samples. We reduce these artifacts by dynamically lowering the variance scale as described in Sec. 3.2. We also illustrate the significance of the suitable jitter sequence and shading filter in Fig. 6. Without these, we can observe that shading is undersampled.

### 5.2 Performance

We compared CPS-T performance against PS and CBR with all images rendered at 1080p resolution by an NVIDIA GTX1080 GPU.
Figure 7: Shading artifacts can be found as a shadow boundary moves downwards (top-row). These artifacts get resolved with PS, but CPS-T still has some errors due to slow convergence (bottom-row).

The Temporal Resolve pass, where TAA and temporal supersampling for CPS-T is performed, takes about 0.8ms in all scenes. While CBR yields a constant 2x reduction in pixel shader invocations over PS, CPS-T leverages 2x2 coarse shading to achieve up to 4x reduction in shading. In our experiments, the shading invocations with CPS-T were significantly less than CBR, as shown in Fig. 9.

However, with thin geometry such as foliage, shading reduction with CPS-T can be degraded due to quad-fragment scheduling of pixel shaders on a GPU. Our MSAA based CPS emulation introduces additional overheads, especially in the z-prepass for fine geometry (see Tab. 1). A native implementation of CPS can avoid this cost.

Overall, the reduction of pixel shading in CPS-T provides a substantial performance benefit, especially when the relative cost of pixel shading is high compared to geometry and screen-space post-processing. This is shown in Tab. 2 for the SPONZA scene, where we emulated higher levels of shader complexity by adding more Forward+ lights to the scene.

5.3 Limitations

Since CPS-T has fewer shading samples per frame compared to CBR, it takes more frames to converge to a full resolution result. This results in a relatively lower shading quality in disoccluded regions and in scenarios with fast motion or dynamic shading, such as the animated shadows in Fig. 7. Another limitation of CPS and CPS-T is that the shading samples are not aligned with visibility samples which results in extrapolation artifacts. This can potentially be addressed by programming the sample positions for an improved spatial distribution of shading and visibility points which we leave for future analysis. We also use a simple disocclusion detection scheme which may fail to report disocclusion events, sometimes even at moderate motion speeds. When this occurs, CPS-T is more likely to expose ghosting artifacts.

6 CONCLUSIONS AND FUTURE WORK

Coarse pixel shading (CPS) can be a useful feature that enables aggressive reduction in shading costs. We demonstrate this by leveraging CPS to derive a temporal supersampling approach with significant performance gains over current techniques like checkerboard rendering, making a strong case for natively supporting this feature in future GPUs.

In the future we would like to improve upon the limitations of our technique by introducing some degree of spatial filtering to overcome the lack of shading samples after a disocclusion event [Schied et al. 2017]. We would also explore to achieve a better spatial distribution of shading and visibility samples inside a single frame and reduce extrapolation artifacts.

ACKNOWLEDGMENTS

We thank Intel’s Advanced Rendering Technology team. We also thank David Blythe and Chuck Lingle for supporting this research. The MiniEngine is courtesy of Microsoft. The B1STRO scene is courtesy of Amazon Lumberyard. The SPONZA scene was modeled by Marko Dabrovic, and SAM1QUEL was modeled by Guillermo M. Leal Llaguno. We are grateful for these scenes. We also appreciate the anonymous reviewers for their insightful comments.

REFERENCES


Figure 8: Quality comparison of different temporal supersampling methods on the SPONZA (top) and SANMIGUEL (bottom) scenes, with some key differences in the results highlighted on the right insets. Row 1: loss of detail in normal maps changes appearance when using CPS. CPS-T closely matches PS and CBR, where the latter also has some jaggy edges. Row 2: CBR loses some visibility on the foliage, but CPS-T displays some Moire artifacts on the curtain. Row 3: loss of thin geometry with CBR (chairs). Row 4: loss of shadow detail with CPS methods.

Figure 9: We plot the total number of PS invocations in the main lighting pass on camera animation sequences. With coarse geometry, CPS(-T) meets its theoretical optimum and reduces the shader invocations close to 25% compared to full PS, while CBR stays constant close to 50%. With complex geometry, quad-fragment shading results in higher PS invocations depending on the camera angle, but in general stays significantly below CBR. Note that the performance gain is not proportional to this number, but becomes more favorable with complex shaders.

https://doi.org/10.1145/3105762.3105770