# Sub-Pixel Shadow Mapping

Pascal Lecocq *
Technicolor

Jean-Eudes Marvie †
Technicolor

Gaël Sourimant ‡
Technicolor

Pascal Gautron §
Technicolor, NVIDIA ARC

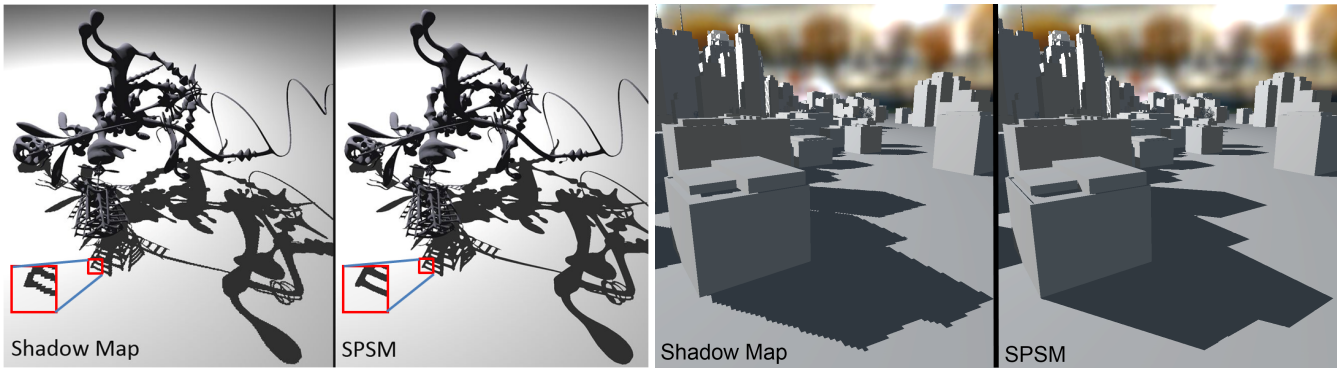



**Figure 1:** *Comparisons of standard Shadow Mapping with Sub-Pixel Shadow Mapping (SPSM). On the left, the YeahRight model (≈ 150 000 polys) using a $1K \times 1K$ shadow map rendered in 11 ms with SPSM. On the right, a large City model (≈ 2 300 000 polys) rendered using a single $2K \times 2K$ shadow map in 41.6 ms. SPSM brings sub-pixel accuracy to shadow maps, avoiding aliasing artifacts while preserving the rendering speed.*

## Abstract

The limited resolution of shadow maps (SM) may result in erroneous shadowing, yielding artificially jagged edges and temporally crawling shadows even using perspective optimization techniques. We introduce Sub-Pixel Shadow Maps (SPSM) for real-time shadow-mapping with sub-pixel precision. Our technique is based on the storage of a fixed-size partial representation of the scene geometry using conservative rasterization, combined with an original reconstruction of shadow edges. With only a small computational overhead our method avoids both perspective and projection aliasing. SPSM reconstructs shadow boundaries with thin details and high temporal consistency even using low resolution shadow maps. Our solution can also be used in conjunction with perspective optimization techniques where temporal and projection aliasing is hardly addressed. SPSM can particularly be used in demanding application such as games and real time rendering engines.

**CR Categories:** I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

**Keywords:** shadows, real-time, shadow-mapping, aliasing

## 1 Introduction

Real time rendering of shadows is an important research topic as it provides essential visual clues for the understandability and realism of 3D scenes. The shadow mapping algorithm [Williams 1978] is a popular technique, widely used in both real-time and production renderers. However, the finite resolution of the shadow map yields aliasing artifacts resulting in erroneous self-shadowing, jagged shadow edges and temporal inconsistencies.

A common approach to overcome aliasing in shadow mapping consists in increasing the resolution of the shadow map. While it does improve shadow quality, it can have huge impact on memory and fillrate consumption, and any close-up on the shadow will inevitably reveal aliasing problems.

In this paper we introduce Sub-Pixel Shadow Mapping (SPSM), a new shadow map-based method generating hard shadows with sub pixel precision and high temporal consistency. SPSM is based on a partial storage of the scene geometry in a single fixed size buffer. The shadow query algorithm uses information stored in the texels and simple geometric projections to build a piecewise linear function accurately representing the shadow boundaries.

Our contributions are:

- A compressed partial geometry representation within a single 128 bits RGBA buffer.
- A per pixel reconstruction method based on simple geometric projections and a tangent estimate, reducing the shadow estimation to evaluating a simple piecewise linear function.
- An inexpensive anti-aliasing method considering shadow multi-sampling as a simple linear function evaluation computed only once per pixel.
- A method to compute more consistent depth regarding the triangle coverage in a texel as an alternative to conservative depth or slope scale-based depth bias methods in conservative rasterization approaches.

## 2 Related works

Many research works have been conducted over the years to address the problem of aliasing with shadow maps. In this section we

*e-mail:pascal.lecocq@technicolor.com
†e-mail:jean-eudes.marvie@technicolor.com
‡e-mail:gael.sourimant@technicolor.com
§e-mail:pgautron@nvidia.com

introduce the works we feel most relevant to our approach. For a concise description of shadowing techniques the reader should refer to [Eisemann et al. 2011] and [Woo and Poulin 2012].

Filtering is a common approach to reduce the perspective aliasing caused by the shadow mapping algorithm. Approaches such as PCF [Reeves et al. 1987] consider the sampling of the shadow depth test to smooth the blockyness of shadows. Variants like VSM [Donnelly and Lauritzen 2006] and CSM [Annen et al. 2007] directly encode a visibility estimate in the shadow map that can be pre-filtered. Efficient filtering is achieved by taking advantage of hardware mipmapping and bilinear interpolation. The filtering approaches tend to blur out the shadow and lose details. These solutions are prone to light and shadow leaking, introducing some disturbing visual artifacts at surface contacts.

Other approaches directly address the source of perspective aliasing by improving the sampling of the shadow map in areas close to the viewer. Warping techniques such as PSM [Stamminger and Drettakis 2002], LiSPSM [Wimmer et al. 2004] or TSM [Martin and Tan 2004] distort the light projection matrix considering post-perspective projections in light space. Other approaches like CSM [Engel 2006] and PSSM [Zhang et al. 2006] consider a partition of the view frustum along viewing axis and assign a separate shadow map to each partition. Such techniques usually require manual tuning of scene dependent parameters. [Lauritzen et al. 2011] introduce a fully automatic partitioning method by analyzing the depth distribution of view samples in light space. Such optimizations drastically reduce the perspective aliasing. Unfortunately, shadow flickering appears during viewpoint motion as the shadow map parameters are constantly updated. Moreover, projective aliasing is more complicated to address with these method.

Perspective aliasing is due to the non linear mapping between shadow map pixels and viewing camera pixels. In *Alias Free Shadow Maps*, [Aila and Laine 2004] proceed to an irregular z-buffer rasterization of the shadow maps based on a hierarchical depth buffer built upon the location of the view samples in projected light space. [Johnson et al. 2005] propose a similar idea but store instead linked lists of view samples in shadow map using dedicated hardware design. These methods were originally implemented on CPU then extended to graphics hardware by [Sintorn et al. 2011] and [Sintorn et al. 2008]. These solutions produce alias-free shadows but rely on a memory-consuming data structure. RTW [Rosen 2012] is an interesting alternative to irregular z-buffer algorithms. The shadow map is rasterized in a rectilinear texture considering non-linear projection on shadow map axes defined by warping maps. Results obtained using 2K RTW shadow maps are close to ray tracing with small computational overhead compared to the standard shadow mapping algorithm. However, adjusting the bias is difficult and has not been addressed in the method. Furthermore, warping maps produce non linear distortions that require either highly tessellated scenes or hardware tessellation.

Perspective aliasing is mostly present at shadows boundaries. Many approaches focusing on shadow boundaries have then been proposed, generally relying on hybrid rendering techniques. [McCool 2000] proceeds to an edge detection by analyzing depth differences in the shadow map. Using a set of reconstruction patterns, the algorithm builds an approximated silhouette mesh and runs a shadow volume algorithm. This method generates $45\,\mathrm{deg}$ edges but does not fully avoid aliasing. Another problem is the fillrate consuming of the stencil shadow volume. In *Shadow Silhouette Maps* [Sen et al. 2003] tackle these problems by computing precise shadow silhouette edges involving mathematics similar to shadow volumes. Silhouette edges are then recorded in a silhouette map using conservative rasterization of quads. The technique effectively improves the perspective aliasing, but relies on heavy silhouette computations and requires an additional map. Also, the silhouette map only stores a single point in each texel, revealing artifacts on strong silhouette curvatures. [Chan and Durand 2004] use the edge detection proposed by [McCool 2000] to identify silhouette pixels in the framebuffer and perform shadow volume rendering only on these pixels using early $z$ rejection. While the solution greatly reduces the overdraw of shadow volumes, the computational overhead remains high compared to standard shadow mapping.

[Hertel et al. 2009] combine conservative rasterization of a shadow map with a GPU ray-tracer to accelerate the rendering of ray-traced shadows. An efficient pixel classification scheme limits ray-tracing operations to a small subset of pixels tagged as uncertain, providing alias-free shadows at interactive framerate. A similar approach have been explored by [Lauterbach et al. 2009] with substantially higher framerates. However, the use of ray-tracing involves dedicated acceleration structures that perform best on static scenes using high end graphic cards.

Closest to our works the RGSM [Dai et al. 2008] propose an alternative storage for the shadow map by encoding the 3D vertex coordinates of the closest visible triangle in each shadow-map texel. Shadowing is then estimated by intersecting rays with the stored in the corresponding texel. The technique addresses both projection and perspective aliasing but suffers from several limitations. First, the method assumes that all triangles are rasterized in the shadow map which limits its use to low polygon geometries or limits the distances from which the shadow map is generated. This issue is potentially compensated by intersecting the triangle stored in nearby texels at the expense of memory bandwidth. This situation is prevalent not only at shadow boundaries but also in lit areas leading to poor performance. Furthermore, storing the 3 vertices coordinates would require to store 9 floats value in a texel. Current hardware generation allow storage of 4 floats or 8 half precision floats in a single RGBA texture map. They do not specify whether they generate one or more render targets to store all the information.

# 3 Sub-Pixel Shadow Maps

## 3.1 Overview

In the spirit of [Dai et al. 2008] we build and store a partial, approximate representation of the scene geometry within the *Sub-Pixel Shadow Map*. Storing triangle information within the shadow map introduces two main challenges. First, most hardware rasterizers generate fragments only if the center of the fragment is covered by the considered triangle. A geometry-based shadow map may thus lack information for texels corresponding to shadow edges, where precision is crucial. The second challenge is the presence of overlaps: in most scenes several triangles are rasterized at a given texel position. As those triangles may not cover the entire texel, each texel should store the list of all overlapping triangles, leading to intractable memory and computational costs.

The *Sub-Pixel Shadow Map* generation enforces the generation of fragments for the entire triangle coverage using *Conservative Rasterization* (Section 3.2). We then simply render the scene into the shadow map, storing geometric and pixel related information for the closest triangle only. We pack the entire triangle definition in SPSM space within a single 128-bit RGBA pixel (Section 3.3), using a novel compression scheme for compact storage of depth derivatives (Section 3.4).

The rendering step accurately estimates shadowing from partial geometry. We project each view sample into light space and proceed to a depth estimation at sub-pixel precision. By leveraging the property of *Conservative Rasterization* and intersecting the tri-

**Figure 2:** *On the left, standard shadow mapping. On the middle our method without silhouette recovery. Right our method with silhouette recovery. Same resolution of shadow map* $(1K \times 1K)$

.

angles stored in the SPSM, we can quickly classify samples as *lit*, *shadowed* or *uncertain*. *Uncertain* samples may be due to the relevant geometry. We address this case by recovering the shadow silhouette from the partial geometry with sub-pixel accuracy (Section 4) as shown in Figure 2.

### 3.2 Conservative rasterization

Accurate shadow determination starts with a consistent recording of triangles responsible for the shadow silhouette in the shadow map. Since these triangles can represent thin geometry structures (tree branches for instance) or geometry viewed from grazing angles, standard rasterization may not capture them if their coverage does not include pixel centers. To tackle this problem we perform a *Conservative Rasterization* of the scene in the sub-pixel shadow map to force the generation of fragments for the entire triangle coverage.

Conservative rasterization introduced by [Hasselgren et al. 2005] guarantees that fragments are produced for any triangle intersecting the texel area. [Hertel et al. 2009] present a similar solution involving slightly different mathematics. They both rely on the same idea: slight forward motion of triangle edges in their normal direction by a length of half a pixel width to force triangle coverage at texel center.

**Conservative Depth.** The generation of consistent depth fragments regarding the conservative rasterization is also a key point for effective capture of silhouette triangles and sub-pixel depth determination. [Hasselgren et al. 2005] propose to store either the maximum or minimum depth within a texel. [Hertel et al. 2009] use a *slope scale depth correction* that shifts triangles towards the light source so that the depth value at texel is minimal. For the SPSM generation we compute the minimum conservative depth as proposed by [Hasselgren et al. 2005] and write it into the depth buffer to ensure the recording of silhouette grazing triangles as described in Figure 3. At the same time, depth at texel center is evaluated and stored in the SPSM. The use of minimum conservative depth may lead to the recording of inconsistent depths, resulting in false occluder detection on surfaces near shadow silhouettes. We avoid this situation in the rendering step by making use of triangle information and depth information stored in the SPSM.

### 3.3 Compact triangle storage

For each texel, the *Sub-Pixel Shadow Map* stores geometric and pixel related information regarding the closest visible triangle. This information is used to quickly determine shadow status on the surface with sub-pixel precision in the rendering pass. The difference with [Dai et al. 2008] is that we keep the triangle storage as small as possible to fit in a single RGBA texel. At the same time we keep

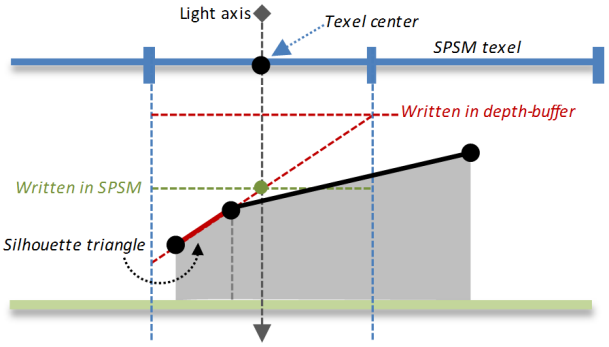



**Figure 3:** *Minimum conservative depth test forces the recording of grazing triangles (in red) responsible for the shadow silhouette. At the same time the depth at texel center is stored in the SPSM to help for further sub-pixel depth evaluation in the rendering pass.*

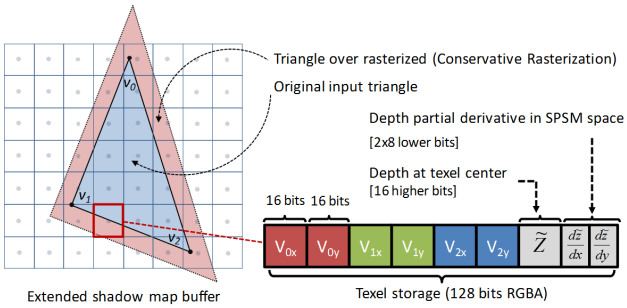



**Figure 4:** *SPSM stores for each texel a compact representation of the closest triangle conservatively rasterized in the shadow map*

sufficient precision to perform precise per pixel shadow determination and maintain high temporal consistency.

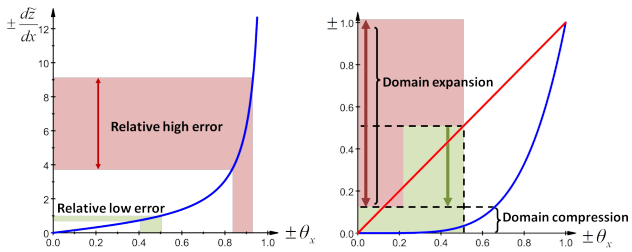
To that end, we represent the triangles by their 2D projections and local variations in SPSM space, and pack each into 128 bits RGBA texel (Figure 4):

- We store the 2D vertex coordinates of the closest triangle in SPSM space, $v_{0 \cdot xy}, v_{1 \cdot xy}, v_{2 \cdot xy}$ as 16 bits packed into the RGB components.
- The depth $\tilde{z}$ at texel center in light space is stored into the 16 higher bits of the Alpha component.
- The depth derivatives $\left(\frac{d\tilde{z}}{dx}, \frac{d\tilde{z}}{dy}\right)$ in SPSM space are stored as quantized $2 \times 8$ bits values in the 16 lower bits of the Alpha component. (Section 3.4)

The compactness of our representation allow us to access triangles stored in the SPSM using a single texture fetch, preserving both the memory occupancy and bandwidth.

**Efficient shadow ray-triangle intersection**. The decomposition of triangle representation as 2D projections and local derivatives provides a faster ray/triangle intersections than the classical way using 3D coordinates. Knowing the location $p$ of a view sample in projected SPSM space, we can faithfully reconstruct the depth on triangle plane using an inexpensive first order linear evaluation:

$$p_{\tilde{z}} = \tilde{z} + (p_x - c_x)\frac{d\tilde{z}}{dx} + (p_y - c_y)\frac{d\tilde{z}}{dy} \qquad (1)$$




(a) Derivative as function of angle (b) Non linear quantization ($x^3$)

**Figure 5:** *Quantization of depth derivatives require higher precision for geometry viewed from grazing angle, impossible to address using a linear quantization (left). We introduce a non linear quantization based on power functions to expand the domain of precision for grazing angle geometries (right)*

This sub pixel depth serves as an early rejection in the intersection test. If the depth test passes, the computation is then reduced to a simple 2D point in triangle test (Algorithm 1).

**Algorithm 1:** Shadow ray/triangle intersection

```
if p_z̃ ≥ p_z then                    /* early rejection test */
    return true;
else
    if p in 2D triangle (v0, v1, v2) then
        return true;
    else
        return false;
```

### 3.4 Derivatives quantization

The 8 bits quantization of depth derivatives require conversion from the unbound domain $[-\infty, +\infty]$ to the bound domain $\left[-\frac{\Pi}{2}, +\frac{\Pi}{2}\right]$. It is achieved by expressing the derivatives as slope angles according to the formula:

$$\theta_{x|y} = \arctan\left(\frac{d\tilde{z}}{dx|y}\right) \tag{2}$$

A naive approach would consider the linear quantization of values in the domain $\left[-\frac{\Pi}{2}, +\frac{\Pi}{2}\right]$ leading to an angular resolution around $0.70\,\text{deg}$. However, a careful analysis of the derivative expression shows that it leads to severe precision issues at grazing angles. As illustrated in Figure 5(a), for angle near domain bounds, corresponding to grazing angle surfaces, a small angular variation represents high derivatives variation. The high precision required in this situation can't be addressed by a linear quantization. Conversely, derivatives vary slowly for low angle i.e almost front facing surfaces. In this case we can afford using low precision values for the quantization.

To improve the angular resolution near domain bounds, we introduce a non linear quantization based on the use of power function $f(x) = x^n$. The power based quantization (see Figure 5(b)) reduces the precision for angular domain around 0 while projecting more values near the domain bounds, leading to an increasing precision.

We found that using the power function $f(x) = x^5$ was sufficient to address most of the projection aliasing issue (Figure 7). The 8 bits quantization achieves resolution of $0.17\,\text{deg}$ for angles around $80\,\text{deg}$, hence providing a $4\times$ accuracy gain.

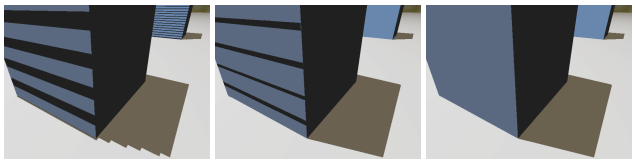


**Figure 6:** *Projection aliasing comparison using a same bias ($1.e^{-5}$). Left, standard shadow mapping without derivatives. Middle, SPSM using linear 8 bits quantized derivatives. Right, SPSM using 8 bits power based quantization (here $x^5$).*

Our derivatives encoding function is then:

$$Q(d) = 127 \times \left(1 + \left(\frac{2}{\pi}\arctan(d)\right)^n\right) \tag{3}$$

Decoding is performed using the inverse of the power function:

$$\frac{d\tilde{z}}{dx|y} = \tan\left(\frac{\pi}{2} \times \left(\frac{Q(d)}{127} - 1\right)^{\frac{1}{n}}\right) \tag{4}$$

In our implementation we pre-compute the 256 possible values of the decoding function into a look-up table for performance.

## 4 Rendering with SPSM

### 4.1 Pixel classification

Similar to [Hertel et al. 2009], we proceed to a quick classification of rendered pixels as *lit*, *shadowed* or *uncertain* considering the properties of the conservative rasterization.

**Lit classification.** For a given point $p$ in the scene, we use Equation 1 to determine whether $p$ lies on the triangle stored in the corresponding SPSM texel. If $p_{\tilde{z}} \approx p_z$, the use of conservative rasterization then ensures that no occluder is present between the light source and $p$. In scenes featuring large lit areas, shadow determination is then obtained at the cost of single texture lookup.

**Shadowed classification.** If $p$ does not belong to the triangle, we intersect the light ray from $p$ with the stored triangle as described in Algorithm 1. While finding an intersection ensures the point is shadowed, the lack of intersection may be due to a missing triangle. A first short-range search is then performed by intersecting the triangles stored in the $3 \times 3$ neighborhood of the texel. Compared to [Hertel et al. 2009] this reduces the number of uncertain pixels only to areas close to casted occluder edges.

**Uncertain classification.** Pixels are classified as uncertain if the previous tests failed to determine the lit status of $p$. The lack of intersection is either due to the absence of occluder or to missing geometry in the SPSM.

### 4.2 Per pixel shadow silhouette recovery

The shadow silhouette operates on uncertain pixels for which all previous queries have failed to determine the lit status of point $p$. Such situation is depicted in Figure 10(a). The central texel is overlapped by the triangles $T2, T3, T4, T6, T7$ and $T8$ and $p$ lies in the

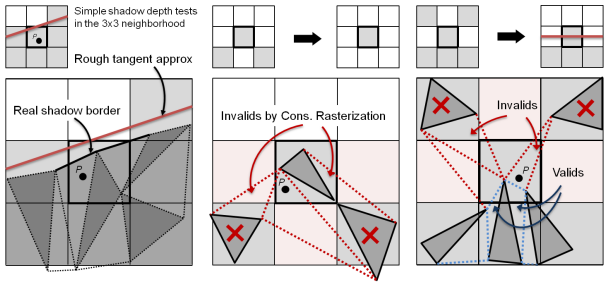



**Figure 7:** *(a) An approximated tangent estimate of the shadow border is obtained by analyzing the depth test pattern in a $3 \times 3$ kernel. (b,c) The Conservative Rasterization of the SPSM guarantees that shadow edges do not cross texels marked in purple. This allows the detection of isolated neighbors that can be either all discarded (b) or not accounted for tangent estimatation (c).*

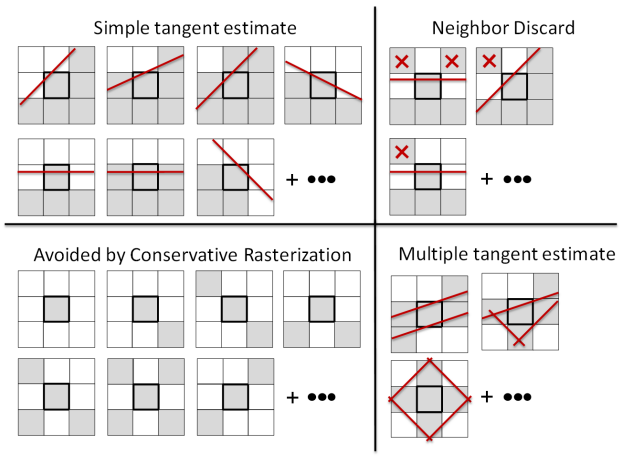



**Figure 8:** *Predetermined pattern set for the tangent estimate, stored in a look-up table. We consider a pattern as a bitmask for direct addressing in the look-up table.*

shadow of $T5$. However only $T0$, $T3$, $T8$ and $T6$ were recorded in texel neighborhood. This results in a false lit status for the point $p$. To overcome this problem, we reconstruct the local shadow silhouette border in shadow map space.

**Tangent estimation of the shadow silhouette.** The recovery process starts with an estimation of an axis representative of the silhouette orientation. Similar to PCF [Reeves et al. 1987], we analyze the result of depth tests in the $3 \times 3$ texel neighborhood (Figure 7(a)). According to the sample distribution, and given predetermined pattern sets (Figure 8), we can roughly estimate the tangent of shadow boundary in central texel. Thanks to the conservative rasterization, pathological case are discarded (see Figure 7(b)). Following the same principle, we can also discard some neighbors for the tangent estimate (Figure 7(c)). More generally, we discard texels which are not 4 neighbor connected. For the case in which all depth test are positive, we assume that an occluder fully covers the area and directly conclude that $p$ is shadowed.

**Consistent depth tests.** The tangent estimation of the shadow silhouette requires consistent depths retrieving from the SPSM. A consistent depth corresponds to the minimum depth spawned by a triangle in the texel area (Figure 9(a)). Unless the triangle fully

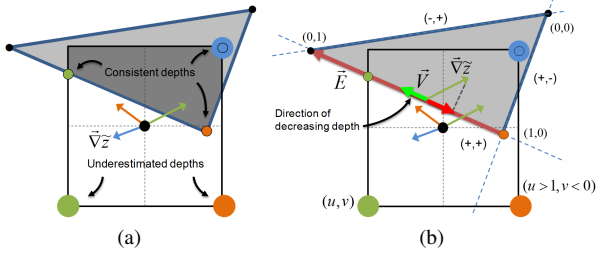



**Figure 9:** *(a) A consistent depth corresponds to the minimum depth spawned by a triangle in the texel area (small colored dots). Conservative depth methods tends to underestimate the minimum depth by considering texel corner instead of triangle coverage (big colored dots). (b) A consistent minimum depth is found using the depth derivatives and simple operations on barycentric $(u, v)$ coordinates.*

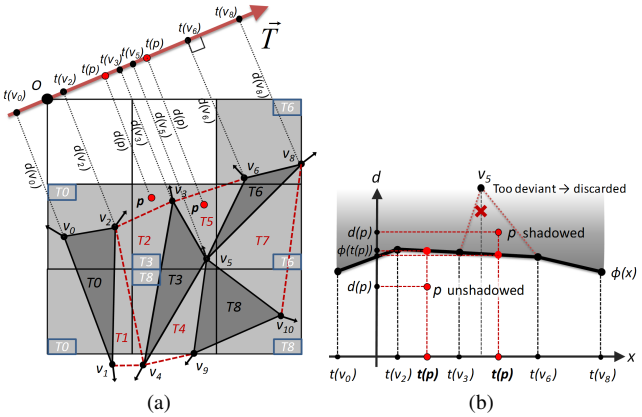



**Figure 10:** *(a) Accurate determination of shadow silhouette is obtained considering selective orthogonal projection of vertices on the tangent silhouette estimate. (b) Selected vertices are linked together to represent a piecewise linear function $\phi(x)$ that recovers the shadow boundary in a plausible way. The lit status of $p$ is simply determined by comparing the orthogonal distance of $p$ with $\phi(x)$ at $x = t(p)$.*

covers the texel, a conservative depth or a slope scale-based depth bias approach will systematically underestimate the depth, resulting in false positive shadow test and erroneous estimation.

Fortunately, using the depth derivatives and simple operations on barycentric coordinates, we can quickly retrieve a consistent minimum depth value (Figure 9(b)). Similar to [Hasselgren et al. 2005], we first identify the texel corner with minimum depth value according to the sign of the depth derivatives $\vec{\nabla}\tilde{z}$. Then, we determine the barycentric coordinate $(u, v)$ of that corner regarding the triangle stored in the texel. If $(u + v) < 1$, the depth is consistent, we then return the depth value at this point according to equation 1. Otherwise, by checking the sign of the $(u, v)$ coordinates, we easily locate the triangle edge $\vec{E}$ that holds the minimum depth. By clamping negatives $u, v$ to 0 and $u, v > 1$ to 1, we identify the triangle vertex with minimum depth in the area. If the vertex lies in the texel, depth at this point is returned. For other cases, we consider the direction of decreasing depth along the edge $\vec{E}$ given by $\vec{V} = -\vec{E} \cdot \vec{\nabla}\tilde{z}$. The consistent minimum depth is then found at the intersection of edge $\vec{E}$ with the texel border in the direction of $\vec{V}$.

**Shadow silhouette recovery.** Once an approximated tangent is found, we perform local projections on that axis to recover the missing information (Figure 10). To that end, we first set the tangent axis origin $O$ on the shadow map corner located at the opposite side of the shadow. The point $p$ is first orthogonally projected on that axis. Position along the axis $t(p)$ and distance $d(p)$ to the axis are computed as follow:

$$\begin{aligned} t(p) &= \vec{Op} \cdot \vec{T} \\ d(p) &= \sqrt{||\vec{Op}|| - t(p)^2} \end{aligned}$$

Then, for each closest triangle, we extract the vertices located on the shadow boundary and discard the other ones. A vertex is considered on the shadow boundary if its normal points toward the tangent axis line. These vertices are then orthogonally projected on the tangent axis and corresponding positions $t(v_i)$ and distances $d(v_i)$ are computed and recorded ( Figure 10(a)). Even though our representation is partial, an accurate silhouette estimate can be obtained by linking these vertices along the projection axis revealing a piecewise linear function $\phi(x)$ (Figure 10(b)). Outlier vertices or vertices located behind a shadow edge are automatically discarded to enforce shadow consistency. The lit status of $p$ can be then easily determined by evaluating $\phi(x)$ according to the position and distance of $p$ as follow:

$$\begin{cases} \phi(t(p)) < d(p), \ p \text{ is shadowed.} \\ \phi(t(p)) \geq d(p), \ p \text{ is lit.} \end{cases}$$

For situations involving multiple tangent estimates (Figure 8 bottom-right), we run the silhouette recovery for each potential tangent axis and take the shadow intersection result.

## 5 Anti-aliasing

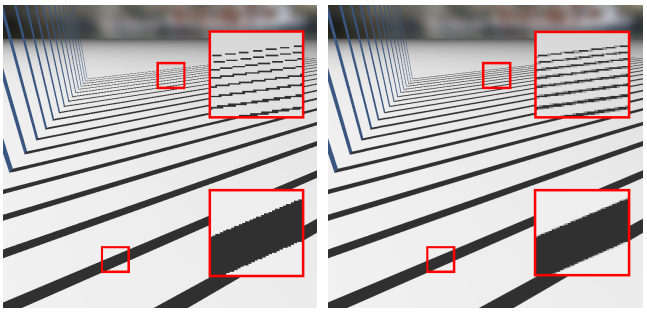
Smoother and visually pleasant images usually rely on anti-aliasing strategies to reduce the pixel-aliasing caused by the finite resolution of the display. MSAA is a common and inexpensive filtering approach based on depth super-sampling but considering a single fragment shading operation. As the result, sharp edges produced by a fragment shader like hard shadows are not filtered and still remain aliased. [Pan et al. 2009] address this issue but the solution relies on a silhouette extraction and the generation of a silhouette map that are computationally expensive.

We propose an inexpensive alternative similar to MSAA that operates multi-sampling only at shadow boundaries, with minimal overhead. We simply reuse the piecewise linear function calculated at once and perform multiple shadow evaluation according to multi-sampling locations in the SPSM. This test is performed only for uncertain pixels, hence reducing the multi-sample shadow operations to a small portion of pixels in the image with a minimal cost (see Figure 11).

## 6 Results

### 6.1 Implementation

We implemented our method using the modified triangle setup as proposed by [Hasselgren et al. 2005] for the conservative rasterization of the SPSM. Triangle-related information is computed at geometry and at fragment shader stages, computing the derivatives using the corresponding built-in functions. Tangent estimates are also precomputed in a table considering the pattern distribution as a bitmask for direct addressing in a look-up table.



(a) SPSM without AA - 9.0 ms (b) SPSM with our AA - 9.3 ms

**Figure 11:** *Shadows anti-aliasing is performed only at shadow boundaries within the fragment shader. Computational cost is minimized by evaluating view samples against the shadow recovery piecewise linear function computed only once.*

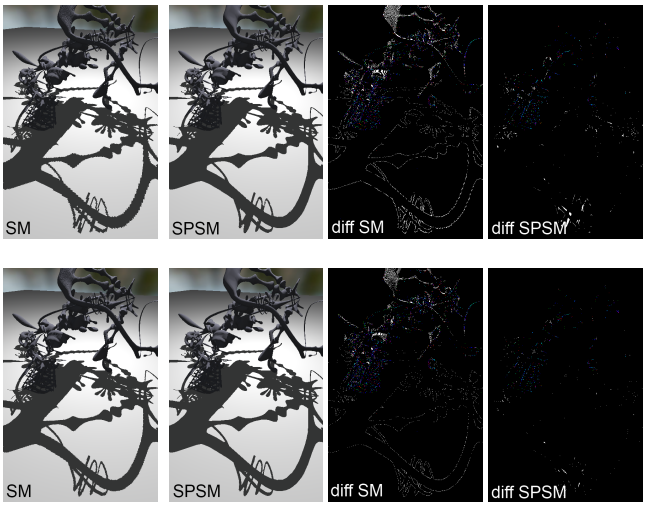



**Figure 12:** *Comparison of standard SM and SPSM with a ray-tracing reference solution using a $1K \times 1K$ shadow map (top) and a $2K \times 2K$ shadow map (bottom).*

### 6.2 Rendering quality

We visually compared the SPSM with the standard shadow mapping on various scenes with different levels of complexity. All images were generated using a single shadow map, without any filtering or perspective optimization techniques. In every scene, the SPSM outperforms the standard shadow mapping algorithm in terms of visual quality. Most of the aliasing observed with the standard shadow mapping disappears with the SPSM, which reveals details impossible to notice with standard shadow maps. Moving the light or varying the shadow map resolution at run time highlights the high temporal consistency of our method (see accompanying video).

We also measured the rendering quality of both standard shadow mapping and SPSM using image differences with a ray-traced shadows solution (Figure 12). The rendering quality achieved by the SPSM is close to ray-tracing solutions with a pixel match of $\approx 99.63\%$ while the standard shadow mapping achieves a pixel match of $\approx 97.93\%$ using a single $2K \times 2K$ shadow map.

Finally, we compared our method with one implementation of the

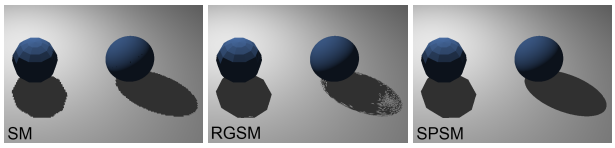



**Figure 13:** *Comparison of standard SM with RGSM ($5 \times 5$ look-ups) and SPSM ($3 \times 3$ look-ups) using a $512 \times 512$ shadow map. The RGSM fails to properly render shadows casted by highly tesselated objects. The SPSM correctly recovers the shadow boundaries of occluders whatever their polygon density.*

| | | Standard SM | | SPSM | |
|---|---|---|---|---|---|
| **Scene** | **#polys** | $1024^2$ | $2048^2$ | $1024^2$ | $2048^2$ |
| Spheres | 12 000 | 5.0 | 6.8 | 6.8 | 8.6 |
| YeahRight | 155 000 | 5.8 | 7.4 | 9.1 | 10.7 |
| City | 2 300 000 | 31.2 | 32.2 | 40 | 41.6 |

**Table 1:** *Rendering times (in ms) for SPSM and standard Shadow Mapping using 1K and 2K resolution shadow maps. Timings include both shadow map generation and scene rendering time.*

RGSM [Dai et al. 2008]. As expected, the RGSM works for low density polygon models but fails to properly render shadows for highly tessellated models (Figure 13 middle), even with large kernel size $\geq 5 \times 5$. Our technique works in any situation, providing consistent shadows whatever the polygon density using only few additional look-ups (Figure 13 right).

### 6.3 Performances

Table 1 shows different timing results for the standard shadow maps and the SPSM. Timings were obtained on a NVIDIA GeForce GTX 580 at a resolution of $1920 \times 1080$. SPSM achieves real time performance with a computational overhead going from $26\%$ to $56\%$ compared to the standard shadow mapping algorithm.

A careful analysis (see Table 2) shows that increasing the resolution of the shadow map reduces the computational overhead of the SPSM. An explanation can be found by analyzing the fragment processing in view space as illustrated in Figure 14. We observe that the costly silhouette recovery concerns few pixels when using high resolution shadow maps. Conversely, using low resolution shadow maps, the area of projected SM texels in view space increases as well as the area of uncertain pixels. In this case, the need for silhouette recovery becomes prohibitive, reaching half the performance of standard shadow mapping. Same critical performance can be reached for scenes featuring a large number of shadow transitions. A better resolution management using perspective optimization techniques could help limit the computational overhead with low resolution shadow maps or high frequency shadows.

| **SM size** | **Standard SM**(ms) | **SPSM**(ms) | **Overhead**(%) |
|---|---|---|---|
| $512^2$ | 5.5 | 10.0 | 81 |
| $1024^2$ | 5.8 | 9.1 | 56 |
| $2048^2$ | 7.4 | 10.7 | 44 |
| $4096^2$ | 14.2 | 18.1 | 27 |

**Table 2:** *SPSM timing overhead measured on the YeahRight model with varying shadow map size.*

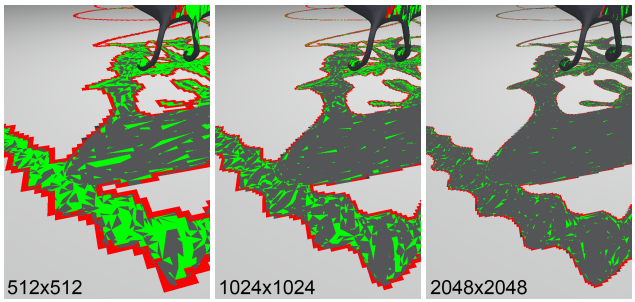



**Figure 14:** *Fragment processing analysis of SPSM in view space with shadow map varying size. Areas marked in green correspond to pixels classified as shadowed using $3 \times 3$ triangle look-ups. Areas marked in red represent pixels for which a silhouette recovery algorithm is performed. Other areas (no color) correspond to pixels classified as lit or shadowed using a single triangle look-up.*

### 6.4 Limitations

While improving the rendering quality, the SPSM shows its own aliasing artifacts due to the limited resolution of the shadow map. First, small holes may be noticed along the shadow border which are due to some triangle inconsistencies between adjacent $3 \times 3$ look-ups. This effect can be counteracted by testing triangles within a larger kernel. An anisotropic kernel can be guided by the tangent estimate to avoid too many look-ups in the SPSM.

We may also observe some shadow fusion if the gap between two occluders in SPSM space is less than the resolution of the shadow map. This also occurs if too many overlapped objects fall in a same area in the SPSM (e.g tree leaves in Figure 15). Increasing the resolution of the SPSM could help reduce these artifacts but at the cost of higher memory occupation and fillrate consumption. Here again, perspective optimization techniques could be considered for better resolution management. A major advantage of SPSM is its easy integration within these techniques.

## 7 Conclusion & Future works

SPSM removes most aliasing artifacts produced by the standard shadow mapping at equivalent resolutions with only a small computational overhead. Our technique recovers accurate shadow boundaries with high temporal consistency. By addressing both the projection and temporal aliasing, SPSM can be beneficial to existing warping or partitioning techniques and can be easily integrated as our method use a single shadow map.

Our improved classification and consistent depth estimation can find also applications in hybrid ray-tracing solutions such as [Hertel et al. 2009]. The depth underestimation, as produced by inclined surfaces for partially covered texels, results in an over classification of pixels as *uncertain* leading to excessive ray-tracing operations. By implementing the consistent depth estimation at fragment level when generating the shadow map, our method could improve the efficiency of these methods by drastically reducing the number of uncertain pixels.

Future work will consider heuristics to avoid shadow fusion, typically using a combination of conservative and non-conservative rasterization. We plan also to investigate soft shadows considering the function $\phi(t)$ as a smooth function. Finally, our compact triangle storage could find application in *Screen Space Ambient Occlusion* techniques for a better estimate of the occlusion factor.

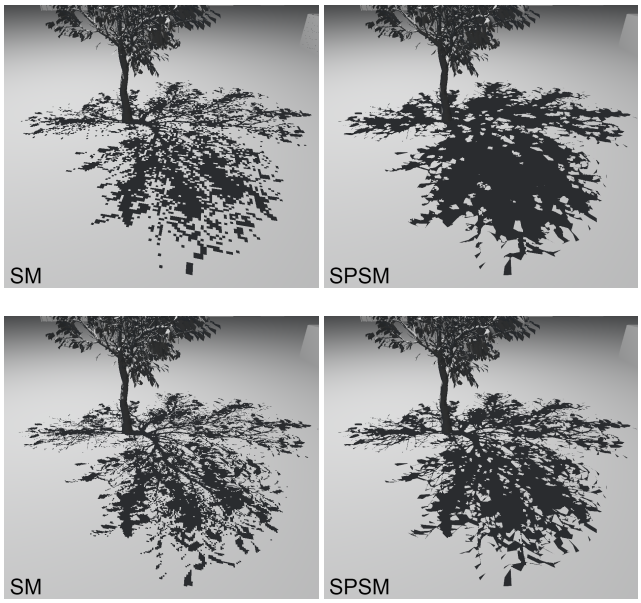



**Figure 15:** *Illustration of the shadow fusion artifact occurring when too many overlapped triangles are projected in the Sub-Pixel Shadow Map (top images, using a* $0.5K$ *shadow map size). Increasing the resolution helps reduce these artifacts (bottom images, using a* $1K$ *shadow map size).*

## References


AILA, T., AND LAINE, S. 2004. Alias-free shadow maps. In *Proceedings of Eurographics Symposium on Rendering 2004*, Eurographics Association, 161–166.

ANNEN, T., MERTENS, T., BEKAERT, P., SEIDEL, H.-P., AND KAUTZ, J. 2007. Convolution shadow maps. In *Rendering Techniques 2007: Eurographics Symposium on Rendering*, Eurographics, Grenoble, France, 51–60.

CHAN, E., AND DURAND, F. 2004. An efficient hybrid shadow rendering algorithm. In *Proceedings of the Fifteenth Eurographics conference on Rendering Techniques*, EGSR'04, 185–195.

DAI, Q., YANG, B., AND FENG, J. 2008. Reconstructable geometry shadow maps. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, ACM, New York, NY, USA, I3D '08, 4:1–4:1.

DONNELLY, W., AND LAURITZEN, A. 2006. Variance shadow maps. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games*, ACM, New York, NY, USA, I3D '06, 161–165.

EISEMANN, E., SCHWARZ, M., ASSARSSON, U., AND WIMMER, M. 2011. *Real-Time Shadows*. A.K. Peters.

ENGEL, W. F. 2006. Cascaded shadow maps. In *ShaderX5: Advanced Rendering Techniques*. Charles River Media, Inc, 197.

HASSELGREN, J., AKENINE-MLLER, T., AND OHLSSON, L. 2005. Conservative rasterization. In *GPU Gems 2*, M. Pharr, Ed. Addison-Wesley, 677–690.

HERTEL, S., HORMANN, K., AND WESTERMANN, R. 2009. A hybrid gpu rendering pipeline for alias-free hard shadows. In *Proceedings of Eurographics 2009*, 59–66.

JOHNSON, G. S., LEE, J., BURNS, C. A., AND MARK, W. R. 2005. The irregular z-buffer: Hardware acceleration for irregular data structures. *ACM Trans. Graph. 24*, 4 (Oct.), 1462–1482.

LAURITZEN, A., SALVI, M., AND LEFOHN, A. 2011. Sample distribution shadow maps. In *Symposium on Interactive 3D Graphics and Games*, ACM, 97–102.

LAUTERBACH, C., MO, Q., AND MANOCHA, D. 2009. Fast hard and soft shadow generation on complex models using selective ray tracing. *Technical Report:TR09-004*.

MARTIN, T., AND TAN, T.-S. 2004. Anti-aliasing and continuity with trapezoidal shadow maps. In *Proceedings of the Fifteenth Eurographics conference on Rendering Techniques*, EGSR'04, 153–160.

MCCOOL, M. D. 2000. Shadow volume reconstruction from depth maps. *ACM Trans. Graph. 19*, 1 (Jan.), 1–26.

PAN, M., WANG, R., CHEN, W., ZHOU, K., AND BAO, H. 2009. Fast, sub-pixel antialiased shadow maps. *Computer Graphics Forum 28*, 7, 1927–1934.

REEVES, W. T., SALESIN, D. H., AND COOK, R. L. 1987. Rendering antialiased shadows with depth maps. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '87, 283–291.

ROSEN, P. 2012. Rectilinear texture warping for fast adaptive shadow mapping. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '12, 151–158.

SEN, P., CAMMARANO, M., AND HANRAHAN, P. 2003. Shadow silhouette maps. *ACM Transactions on Graphics - Proceedings of SIGGRAPH 2003 22*, 3, 521–526.

SINTORN, E., EISEMANN, E., AND ASSARSSON, U. 2008. Sample-based Visibility for Soft Shadows Using Alias-free Shadow Maps. *Computer Graphics Forum (Proceedings of the Eurographics Symposium on Rendering 2008) 27*, 4, 1285–1292.

SINTORN, E., OLSSON, O., AND ASSARSSON, U. 2011. An efficient alias-free shadow algorithm for opaque and transparent objects using per-triangle shadow volumes. In *ACM SIGGRAPH Asia 2011*, SIGGRAPH Asia 2011.

STAMMINGER, M., AND DRETTAKIS, G. 2002. Perspective shadow maps. In *ACM Transactions on Graphics*, 557–562.

WILLIAMS, L. 1978. Casting curved shadows on curved surfaces. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '78, 270–274.

WIMMER, M., SCHERZER, D., AND PURGATHOFER, W. 2004. Light space perspective shadow maps. In *Proceedings of the Fifteenth Eurographics conference on Rendering Techniques*, EGSR'04, 143–151.

WOO, A., AND POULIN, P. 2012. *Shadow Algorithms Data Miner*. CRC Press.

ZHANG, F., SUN, H., XU, L., AND LUN, L. K. 2006. Parallel-split shadow maps for large-scale virtual environments. In *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, ACM, New York, NY, USA, VRCIA '06, 311–318.