Volume xx (200y), Number z, pp. 1–16

Revectorization-Based Soft Shadow Mapping

M. C. F. Macedo, A. L. Apolinário Jr., and K. A. Agüero

Department of Computer Science, Federal University of Bahia (UFBA), Bahia, Brazil
marciocfmacedo@gmail.com, antonio.apolinario@ufba.br, kaguero@ufba.br

Abstract

In this paper, we present revectorization-based soft shadow mapping, an algorithm that enables the rendering of visually plausible anti-aliased soft shadows in real time. In revectorization-based shadow mapping, shadow silhouettes are anti-aliased and filtered on the basis of a discontinuity space. By replacing the filtering step of the theoretical framework of the percentage-closer soft shadow algorithm by a revectorization-based filtering algorithm, we are able to provide anti-aliasing mainly for near contact shadows or small penumbra sizes generated from low-resolution shadow maps. Moreover, we present a screen-space variant of our technique that generates visually plausible soft shadows with an overhead of only $\sim$ 23% in processing time, when compared to the fastest soft shadow algorithms proposed in the literature, but that introduces shadow overestimation artifacts in the final rendering.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1. Introduction

Shadow rendering is an essential feature for many computer graphics applications because shadows provide important visual cues about the scene. However, the real-time computation of high-quality, accurate shadows is still a challenging problem, mostly because this kind of rendering remains too expensive to be done interactively and dynamically.

Shadow mapping [Wil78] is commonly used to compute real-time hard shadows. In this technique, the scene is totally illuminated or shadowed by a point light source. In the real world, such hard shadows rarely occur, because they do not take into account the penumbra effect produced by the partial occlusion of the light source. Several techniques [RSC87, DL06, AMS*08, LM08, Sal08, PK15] have proposed the use of filtered hard shadows to fake the penumbra effect. Even in this case, the shadows are not that realistic because these techniques compute fixed-size penumbras at the shadow silhouette. However, in the real world, the penumbra size varies according to the distance from the light source to both light blocker and shadow receiver objects.

Percentage-Closer Soft Shadows (PCSS) [Fer05] is a well-known extension of shadow mapping to compute soft shadows in real time. It takes into account the effect produced by area light sources to generate visually plausible soft shadows. The method is simple to implement and provides real-time performance for small area light sources. On the other hand, PCSS produces aliasing artifacts for large area light sources, is not scalable for large filter sizes and does not produce accurate soft shadows, since the technique works on the basis of a single point light source [ESAW11].

Many approaches have been proposed to solve the problems of scalability and aliasing of PCSS (see Section 2). The most successful ones typically pre-filter the shadow map, achieving high-quality filtering at constant time, but introduce light leaking, an artifact that erroneously renders a fully shadowed region as a lit one. Despite this fact, pre-filtering techniques still produce aliasing under low-resolution shadow maps or small penumbra sizes.

To reduce the aforementioned problems, we introduce the Revectorization-Based Soft Shadow Mapping (RBSSM) algorithm. We extend Revectorization-Based Shadow Mapping (RBSM) [MA16] to produce real-time anti-aliased visually plausible soft shadows on the basis of the PCSS framework and a single point light source. Besides, we propose a screen-space variant of RBSSM, Screen-Space RBSSM (SSRBSSM), to gain performance, while keeping both the shadow anti-aliasing and light leaking reduction of RBSSM.

In this context, our main contributions are twofold: **1.** The use of the shadow revectorization for anti-aliased soft shadow rendering that is mainly useful for near contact shadows or small penumbra sizes generated from low-resolution shadow maps; **2.** The proposition of two algorithms that compute revectorization-based soft shadows on the basis of light and camera spaces, respectively. Even by simulating soft shadows from a single point light source, the former technique is able to produce soft shadows closer to the ground-truth ones, even for complex scenarios. Meanwhile, the latter is able to generate soft shadows about two to four times faster than the light-space approach, at the cost of introducing more shadow overestimation.

2. Related Work

In this section, we review the most relevant approaches related to our solution. For a complete review of the existing soft shadow algorithms, the reader should refer to the books [ESAW11, WP12].

Percentage-Closer Soft Shadows: To produce visually plausible soft shadows, the PCSS technique uses an assumption that light source, blocker and receiver objects are all planar and parallel to each other [SS98]. Then, by the use of similar triangles constructed on the basis of the average distance of the blocker objects to the light source, the light source size and the receiver depth, a variable penumbra size can be estimated and filtered using Percentage-Closer Filtering (PCF) [RSC87]. Despite the advantages of PCSS, such as ease of implementation and the use of only one shadow map to produce soft shadows, the technique scales poorly for large filter sizes and produces aliased shadows [ESAW11] mainly for large area light sources.

Aiming to solve the aliasing problem of PCSS, Euclidean Distance Transform Soft Shadow Mapping (EDTSSM) [MA17a] is a technique that computes visually plausible soft shadows as an Euclidean Distance Transform over the hard shadows generated by RBSM. Indeed, EDTSSM provides suitable anti-aliasing for PCSS, at the cost of shadow overestimation artifacts that are introduced in the scene. Moreover, EDTSSM is slightly slower than PCSS because of the EDT computation.

Back-Projection: Alternatively to PCSS, back-projection techniques unproject a micropatch for each shadow map texel and use this geometric representation to compute the amount of the light source occluded by the blocker objects [GBP06, BCS08]. Micropatches are approximations of the blocker geometry. Thus, they may cause shadow overestimation and light leaking. The authors of [SS07, GBP07, YFGL09] propose different solutions to reduce such problems and generate high-quality soft shadows using a single shadow map. However, these methods are still prone to artifacts and achieve only interactive performance.

Pre-Filtering: To solve both problems of PCSS, pre-filtering techniques commonly use a filterable function to approximate the average blocker depth estimation and/or the shadow test.

Summed-Area Variance Shadow Mapping (SAVSM) [Lau08] replaces the PCF step of the PCSS framework by the Variance Shadow Mapping (VSM) [DL06]. Convolution Soft Shadow Mapping (CSSM) [ADM*08] extends the theory behind the Convolution Shadow Mapping (CSM) [AMS*08] to propose a constant-time average blocker depth estimation on the basis of a pre-filtered shadow map and pre-filtered Fourier series basis images. Variance Soft Shadow Mapping (VSSM) [DY10, YDF*10] estimates the average blocker depth efficiently on the basis of a pre-filtered shadow map and the Chebyshev's Inequality. Exponential Soft Shadow Mapping (ESSM) [SFY13] uses the Exponential Shadow Map (ESM) [AMS*08, Sal08] theory to estimate the average blocker depth and compute the final soft shadow intensity in constant time on the basis of a pre-filtered exponential shadow map. Similarly to VSSM, Moment Soft Shadow Mapping (MSSM) [PMWK16] generates a pre-filtered moment shadow map [PK15], and uses it to estimate the average blocker depth and perform the shadow filtering by solving the Hamburger moment problem.

Different from the previous approaches, the work proposed in [SDMS15] uses a new pre-filtering method in which a per-texel fragment list of all blockers is stored in a multi-layer shadow map [XTP07]. Liktor et al. [LSMD15] samples a 4D shadow light field using a stochastic soft shadow map and converts such samples to a pre-filterable basis.

Pre-filtering techniques produce visually plausible soft shadows with constant-time filtering. Most of them make use of Summed-Area Tables (SAT) [Cro84] as their pre-filtering function to avoid the brute-force sampling proposed by PCSS. The drawback shared by the pre-filtering techniques is that they are prone to light leaking or incorrect shadow computation at contact borders, making their use unsuitable for complex scenarios.

Screen-Space Filtering: An alternative to improve the performance of the soft shadow computation at the cost of lower visual accuracy is to perform some or all the soft shadow filtering in the screen space. Screen-Space Percentage-Closer Soft Shadows (SSPCSS) [MKHS10] proposes that the PCSS framework must be fully computed in screen space by the use of a separable cross bilateral filter [PvV05]. Screen-Space Anisotropic Blurred Soft Shadows (SSABSS) [ZS11] computes the average blocker depth in light space and performs a screen-space anisotropic Gaussian blur over the penumbra size to improve the visual quality of the SSPCSS algorithm. Buades *et al.* [BGC15] proposes a separable algorithm to estimate the average blocker depth and uses Gaussian filtering to filter the shadows in screen space. Their approach generates visually plausible soft shadows on the basis of user-defined parameters.

In our proposal, rather than coupling the basis of PCSS with the anti-aliasing provided by other techniques, such as [LMSG14, WHL15], we use the RBSM [MA16] because this technique is able to produce anti-aliased hard and filtered hard shadows with only $\sim 1.2\%$ of additional processing time when compared to shadow mapping [MTAA17]. Then, we replace the PCF step of PCSS by such a revectorization-based filtering in the shadow map space to produce anti-aliased real-time soft shadows, even for low-resolution shadow maps and small filter sizes.

The concept of shadow revectorization has already been used to compute accurate soft shadows without achieving real-time frame rates [MA17b], on the basis of an adaptive area light source sampling. Here, we aim to make a novel use of shadow revectorization, proposing new solutions to compute real-time, visually plausible soft shadows on the basis of a single point light source. Hence, we generate less accurate shadows than techniques that sample the area light source to compute accurate soft shadows (such as [MA17b]), but we provide real-time performance for the shadow rendering.

3. Shadow Revectorization

RBSM [MA16] is a technique that uses the available image resolution to reduce the jagged pattern of aliased shadows generated by shadow mapping when using low-resolution shadow maps. An overview of RBSM can be seen in Figure 1 and the pseudocode is shown in Algorithm 1. Figure 1 already shows both camera and light spaces perspectively aligned just for the sake of better legibility. In this section, we show how RBSM is general enough to handle distinct scene configurations, aligning and taking into account

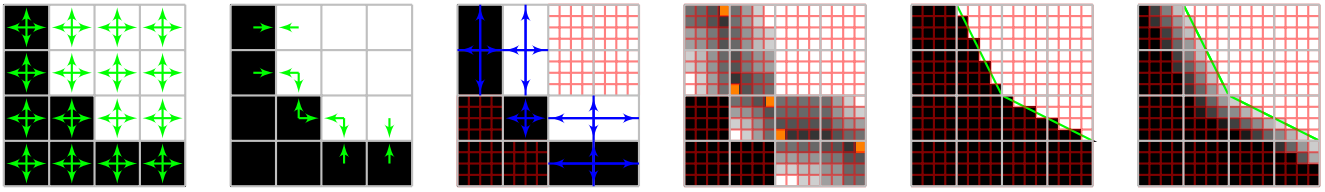


Figure 1: *The main steps of the revectorization process. The picture illustrates a global view of the process, but, in fact, each step runs in parallel for every camera-view fragment. (a) The shadow map S allows the evaluation of the shadow test V_{SM} (black and white square filling) for each S texel (gray grid) and the difference **D** of V_{SM} for each 4-connected neighbourhood (green arrows). (b) Distinct shadow tests between neighbour S texels (green arrows) indicate the presence of shadow silhouettes, transitions between lit and umbra fragments in the camera space. (c) Traversal directions (blue arrows) defined by the projection of camera-view fragments (red grid) in S allow the estimation of each shadow silhouette size in the camera space. (d) The normalized distance (gray shades) of each camera-view fragment to the aliased shadow silhouette; orange squares represent lit and shadowed camera-view fragments with the smallest distance to the shadow silhouette. (e) The new visibility condition of each camera-view fragment is estimated according to its distance to the shadow silhouette, defining a revectorized hard silhouette (green line). (f) The revectorized filtered hard shadow rendering. In this picture, the camera (red grid) and light (gray grid) spaces are aligned and perspectively undistorted for better legibility of the revectorization process.*

Algorithm 1 Revectorization-based shadow mapping

```
1:  for each frame do
2:      S ← RENDERSHADOWMAP;
3:      for each camera-view fragment p in parallel do
4:          p̃ ← TRANSFORMTOLIGHTSPACE(p);
5:          V_SM ← COMPUTESHADOWTEST(p̃, S);
6:          D ← COMPUTESILHOUETTEDIRECTION(V_SM);
7:          if D ≠ {0,0,0,0} then
8:              r ← ESTIMATERELATIVEPOSITION(p̃, S, V_SM);
9:              V_RBSM ← PERFORMANTIALIASING(r, V_SM);
10:         else
11:             V_RBSM ← V_SM;
12:         end if
13:     end for
14: end for
```

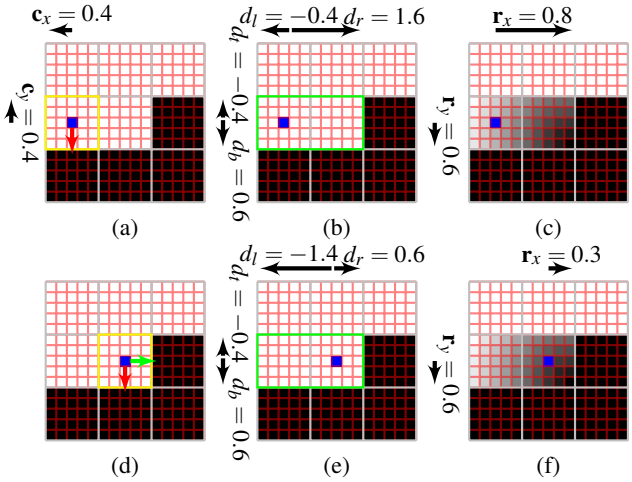


Figure 2: *Let a sampled camera-view (red grid) fragment $\boldsymbol{F}_1$ (blue square in (a)) whose coordinates when projected in a shadow map texel (yellow square in (a)) is **c**. From **c**, we can compute signed distances (d_l, d_r, d_t, d_b) of $\boldsymbol{F}_1$ to the shadow silhouette (green rectangle in (b)) and estimate the relative position **r**, in parallel, for every camera-view fragment (c). Let $\boldsymbol{F}_2$ (blue square in (d)) be a neighbour of $\boldsymbol{F}_1$. If both share at least a single vertical silhouette direction (red arrow in (d)), we can use the step size to update the horizontal signed distances previously estimated (e) and compute the relative position of $\boldsymbol{F}_2$ in the shadow silhouette (f).*

the perspective projection in the images generated from camera and light spaces.

Let us assume **p** as a surface point in the camera view and $\tilde{\mathbf{p}}$ as **p** transformed into the light source view. Also, let S be a shadow map with m rows and n columns, where each pixel $\mathsf{S}(i,j) \in [0,1]$, with $i \in [0,n]$ and $j \in [0,m]$. S stores the distance $\tilde{\mathbf{p}}_z$ of the **closest** fragment $\tilde{\mathbf{p}}$ seen from the light source and projected in the shadow map texture coordinates (i,j). We define the binary shadow mapping test $V_{\mathrm{SM}}(\tilde{\mathbf{p}}_z, \mathsf{S}(i,j)) \in \{0,1\}$ [Wil78], or simply V_{SM}, equals to 0 if $\tilde{\mathbf{p}}_z > \mathsf{S}(i,j)$ and 1 otherwise. $V_{\mathrm{SM}} = 0$ means that the fragment **p** is in umbra (black squares of Figure 1-(a)), while $V_{\mathrm{SM}} = 1$ means that **p** is lit (white squares of Figure 1-(a)).

The shadow test V_{SM} is known to generate aliased shadows. To detect if **p** is located at a transition between umbra and lit sides, we compute **D**, a 4D binary vector, representing the difference of V_{SM} between $\tilde{\mathbf{p}}$ and its 4-connected neighbours (Line 6 of Algorithm 1, Figure 1-(b)). For shadow silhouette fragments, **D** has at least one non-zero component. **D** can be measured by means of comparisons or by the absolute difference of neighbour shadow tests.

$$\begin{aligned}\mathbf{D}(i,j) = (&|V_{\mathrm{SM}}(\tilde{\mathbf{p}}_z, \mathsf{S}(i-1,j)) - V_{\mathrm{SM}}(\tilde{\mathbf{p}}_z, \mathsf{S}(i,j))|,\\ &|V_{\mathrm{SM}}(\tilde{\mathbf{p}}_z, \mathsf{S}(i+1,j)) - V_{\mathrm{SM}}(\tilde{\mathbf{p}}_z, \mathsf{S}(i,j))|,\\ &|V_{\mathrm{SM}}(\tilde{\mathbf{p}}_z, \mathsf{S}(i,j-1)) - V_{\mathrm{SM}}(\tilde{\mathbf{p}}_z, \mathsf{S}(i,j))|,\\ &|V_{\mathrm{SM}}(\tilde{\mathbf{p}}_z, \mathsf{S}(i,j+1)) - V_{\mathrm{SM}}(\tilde{\mathbf{p}}_z, \mathsf{S}(i,j))|),\end{aligned} \tag{1}$$

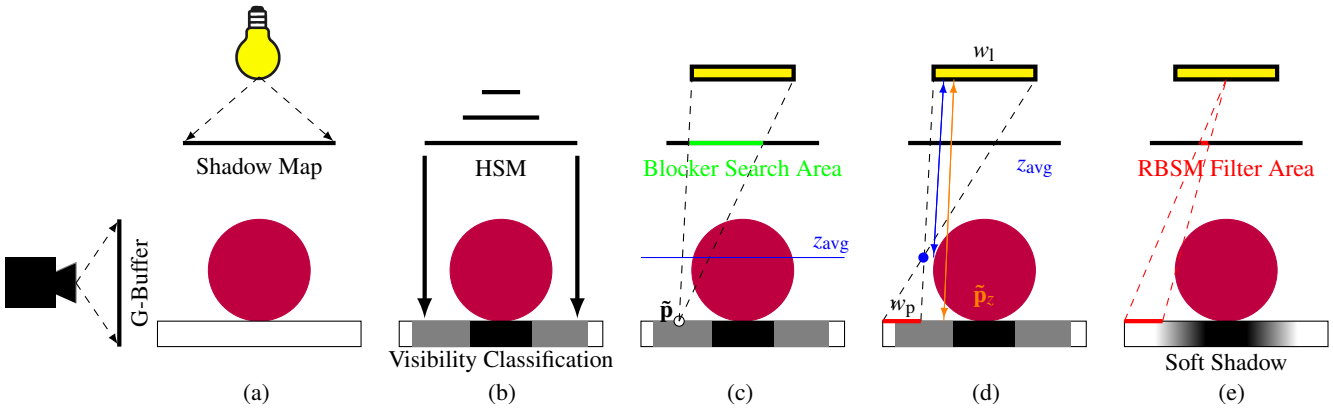


Figure 3: *An overview of RBSSM. First, shadow map and G-buffer are rendered from light and camera spaces, respectively (a). Then, a hierarchical shadow map (HSM) is built to locate potential penumbra fragments in the camera view (b). For these fragments, the average blocker depth (z_{avg} in (c)) is computed, the penumbra width (w_p in (d)) is estimated and filtered using RBSM to generate soft shadows (e).*

The next step of RBSM consists in the estimation of the 2D normalized relative position $\mathbf{r}$ (Line 8 of Algorithm 1 and Figure 2-(c)). For each fragment in the shadow silhouette (Line 7 of Algorithm 1), $\mathbf{r}$ stores the distances, normalized to the unit interval, of those fragments to the shadow silhouettes located at horizontal ($\mathbf{r}_x$) and vertical ($\mathbf{r}_y$) directions. Let $\mathbf{c}$ be the normalized 2D coordinates of a camera-view fragment in its corresponding shadow map texel that takes into accout the perspective projection effect (Figure 2-(a))

$$\mathbf{c} = (\tilde{\mathbf{p}}_x n - \lfloor \tilde{\mathbf{p}}_x n \rfloor, \tilde{\mathbf{p}}_y m - \lfloor \tilde{\mathbf{p}}_y m \rfloor), \tag{2}$$

where n and m are the shadow map width and height. Let d_l, d_r, d_t and d_b (Figure 2-(b)) be four floating-point signed distances computed as a sum of $\mathbf{c}$ ($\mathbf{c}_x$ for d_l, 1 - $\mathbf{c}_x$ for d_r, $\mathbf{c}_y$ for d_t and 1 - $\mathbf{c}_y$ for d_b) and the number of **additional** shadow map texels accessed before the shadow map traversal ending for left, right, top and bottom directions. The signs of these distances are given by the condition of the shadow map traversal ending. Every shadow map traversal (Figure 1-(c)) begins at the shadow map texel where the fragment is located and ends at the shadow map texel whose shadow test is different from the one initially estimated (positive sign) or when the traversal steps out ($\mathbf{D} = \{0,0,0,0\}$) of the shadow silhouette (negative sign), as shown in Figure 2-(b). Then, $\mathbf{r}$ is estimated as[†]

$$\mathbf{r} = \left(\frac{\max(d_l, d_r)}{|d_l| + |d_r|}, \frac{\max(d_t, d_b)}{|d_t| + |d_b|}\right). \tag{3}$$

The line equation that defines the revectorized silhouette is given by $\mathbf{r}_y = -\mathbf{r}_x + 1$. Then, a fragment must be put in shadow if it is between the revectorized and the aliased silhouette (*i.e.,* $\mathbf{r}_y < -\mathbf{r}_x + 1$). Hence, the hard shadow visibility function of RBSM (Line 9 of Algorithm 1, Figure 1-(e)) is defined as

$$V_{\text{RBSM}}(V_{\text{SM}}, \mathbf{r}) = \begin{cases} 0 & \text{if } (V_{\text{SM}} = 0) \vee (\mathbf{r}_y < 1 - \mathbf{r}_x), \\ 1 & \text{otherwise.} \end{cases} \tag{4}$$

[†] For more details on the inner workings of the function **estimateRelativePosition** (Line 8 of Algorithm 1), we refer the reader to see the Section 1 of the supplementary document.

Algorithm 2 Revectorization-based soft shadow mapping

```
1:  w_l ← light source size;
2:  w_k^avg ← 1D number of samples to estimate z_avg;
3:  for each frame do
4:      S ← RENDERSHADOWMAP;
5:      G ← RENDERGBUFFER;
6:      H ← BUILDHIERARCHICALSHADOWMAP;
7:      for each sample p in G in parallel do
8:          p̃ ← TRANSFORMTOLIGHTSPACE(p);
9:          V_HSM ← ESTIMATEVISIBILITYFROMHSM(p̃, H);
10:         if V_HSM = umbra or V_HSM = lit then
11:             V_RBSSM ← V_HSM
12:         else
13:             z_avg ← COMPUTEAVGBLOCKDEPTH(w_k^avg, p̃, S);
14:             w_p ← ESTIMATEPENUMBRASIZE(w_l, z_avg, p̃_z);
15:             V_RBSSM ← COMPUTESOFTSHADOW(w_p, p̃, S);
16:         end if
17:     end for
18: end for
```

RBSM can produce filtered hard shadows (Figure 1-(f)) with a smooth intensity transition that goes from maximum (1) at the anti-aliased lit-sided silhouette, to minimum (0), at the anti-aliased shadow-sided silhouette. Since the line equation of RBSM is $\mathbf{r}_x + \mathbf{r}_y = 1$, we can define $V_{\text{RBSM}} = \mathbf{r}_x + \mathbf{r}_y$ for lit fragments ($V_{\text{SM}} = 1$), and $V_{\text{RBSM}} = 1 - (\mathbf{r}_x + \mathbf{r}_y)$ for shadowed fragments ($V_{\text{SM}} = 0$), so

$$V_{\text{RBSM}}(V_{\text{SM}}, \mathbf{r}) = (V_{\text{SM}})(\mathbf{r}_x + \mathbf{r}_y) + (1 - V_{\text{SM}})(1 - \mathbf{r}_x - \mathbf{r}_y). \tag{5}$$

RBSM is able to perform anti-aliasing by the revectorization of hard shadows. In this paper, we propose the extension of RBSM to generate real-time anti-aliased visually plausible soft shadows.

4. Revectorization-Based Soft Shadow Mapping

RBSSM computes anti-aliased shadows based on PCSS, a hierarchical shadow map (HSM), and RBSM. The PCSS framework is

used for soft shadow filtering. A HSM speeds up soft shadow rendering by locating potential penumbra fragments. Finally, RBSM adds the anti-aliasing effect to the soft shadows. Each one of these major steps is described in detail as follows.

Percentage-Closer Soft Shadows: To estimate the penumbra width of a given region, PCSS proposes that one needs to first estimate the average blocker depth $z_{\text{avg}}(w_{\text{k}}^{\text{avg}}, \tilde{\mathbf{p}}, \mathsf{S})$, or simply z_{avg}. In our implementation, we use a square kernel with $w_{\text{k}}^{\text{avg}} \times w_{\text{k}}^{\text{avg}}$ samples

$$z_{\text{avg}} = \frac{\sum_{i=-\text{A}}^{\text{A}} \sum_{j=-\text{A}}^{\text{A}} (1 - V_{\text{SM}}) \mathsf{S}(\tilde{\mathbf{p}}_x + i r_{\text{avg}}, \tilde{\mathbf{p}}_y + j r_{\text{avg}})}{\varepsilon + \sum_{i=-\text{A}}^{\text{A}} \sum_{j=-\text{A}}^{\text{A}} (1 - V_{\text{SM}})}, \quad (6)$$

where $\varepsilon = 0.001$, $\text{A} = \lfloor w_{\text{k}}^{\text{avg}}/2 \rfloor$, $r_{\text{avg}} = w_{\text{l}}(\tilde{\mathbf{p}}_z - z_{\text{near}}) \,/\, (\tilde{\mathbf{p}}_z w_{\text{k}}^{\text{avg}})$ is the blocker search region step size [MFB08], z_{near} is the shadow map's near plane, and w_{l} is the light source width.

Based on the parallel-planar assumption of PCSS, the variable penumbra width $w_{\text{p}}(w_{\text{l}}, z_{\text{avg}}, \tilde{\mathbf{p}}_z)$, or simply w_{p}, can be estimated as [Fer05]

$$w_{\text{p}} = w_{\text{l}}(\tilde{\mathbf{p}}_z - z_{\text{avg}}) \,/\, z_{\text{avg}}, \quad (7)$$

and projected onto the near plane of the shadow map [ESAW11].

To transform hard into soft shadows, PCSS uses the PCF algorithm [RSC87] to compute the penumbra intensity of each fragment as the result of a convolution filter applied to the hard shadow tests measured by shadow mapping. This filter has width of w_{p} and is sampled by a kernel of $w_{\text{k}} \times w_{\text{k}}$ samples and step size $\frac{w_{\text{p}}}{w_{\text{k}}}$. To make PCSS faster, we use a G-buffer [ST90] of viewport width w and height h, called G, that allows, for each pixel $\mathsf{G}(i, j)$, the recovery of the world-space position $\mathbf{p}$ of the visible surface points in the camera viewpoint. Also, we build a HSM to further identify potential visible fragments located in the penumbra, restricting the use of PCSS to the visible fragments in the camera space that are potentially located in penumbra.

Hierarchical Shadow Map: Let H be a two-channel pyramidal-like min-max hierarchical shadow map texture [GBP06], that, at level l, has $\frac{m}{2^l}$ rows and $\frac{n}{2^l}$ columns. Each pixel in the first level ($l = 0$) of H is defined by the shadow map itself. Then, the subsequent levels of H can be iteratively built similarly to mip-mapping, but by computing both minimal and maximal values of the depths stored in a 2×2 region of the previous level.

With the map H, given the shadow map region intersected by the frustum formed by the light source area and the surface point $\tilde{\mathbf{p}}$, the depth values $z_{\text{min}} \in [0, 1]$ and $z_{\text{max}} \in [0, 1]$ of the corresponding HSM level must be retrieved to allow the evaluation of the illumination condition of the surface point by the visibility function $V_{\text{HSM}}(\tilde{\mathbf{p}}_z, z_{\text{min}}, z_{\text{max}})$ [GBP06]

$$V_{\text{HSM}}(\tilde{\mathbf{p}}_z, z_{\text{min}}, z_{\text{max}}) = \begin{cases} \text{lit} & \text{if } \tilde{\mathbf{p}}_z \leq z_{\text{min}}, \\ \text{in umbra} & \text{else if } \tilde{\mathbf{p}}_z \geq z_{\text{max}}, \\ \text{in penumbra} & \text{otherwise.} \end{cases} \quad (8)$$

A HSM is useful to speed up soft shadow rendering, but PCSS is still prone to generate aliased soft shadows due to PCF. In RBSSM, we extend PCSS to provide filtering over revectorized hard shadows (Figure 1-(e)), generating anti-aliased soft shadows.

Algorithm 3 Revectorization-based soft shadow filtering

```
1:  procedure COMPUTESOFTSHADOW(w_p, p̃, S)
2:      w_k ← 1D number of samples for soft shadow filtering;
3:      V_RBSSM ← 0;
4:      D^prev ← null; s̃^prev ← null;
5:      for y from −⌊w_k/2⌋ to ⌊w_k/2⌋ serially do
6:          for x from −⌊w_k/2⌋ to ⌊w_k/2⌋ serially do
7:              s̃ = {p̃_x + x w_p/w_k, p̃_y + y w_p/w_k, p̃_z};
8:              V_SM ← COMPUTESHADOWTEST(s̃, S);
9:              D ← COMPUTESILHOUETTEDIRECTION(V_SM);
10:             if SAMEDIRECTION(D, D^prev) ∧ s̃_y = s̃_y^prev then
11:                 if d_r > 0 then
12:                     d_l ← d_l − w_p/w_k;
13:                     d_r ← d_r − w_p/w_k;
14:                 else
15:                     d_l ← d_l + w_p/w_k;
16:                     d_r ← d_r + w_p/w_k;
17:                 end if
18:             else
19:                 {d_l, d_r, d_t, d_b} ← COMPUTEDISTS(s̃, S, V_SM);
20:             end if
21:             r ← { max(d_l,d_r)/(|d_l|+|d_r|), max(d_t,d_b)/(|d_t|+|d_b|) };
22:             V_RBSM ← PERFORMANTIALIASING(r, V_SM);
23:             V_RBSSM ← V_RBSSM + V_RBSM;
24:             D^prev ← D; s̃^prev ← s̃;
25:             x ← x + 1;
26:         end for
27:         y ← y + 1;
28:     end for
29:     return V_RBSSM / w_k^2;
30: end procedure
```

Soft Shadow Revectorization: An overview of RBSSM is shown in Figure 3 and in Algorithm 2. First, we generate the shadow map S, the G-buffer G, and the hierarchical shadow map H (Lines 4-6 of Algorithm 2 and Figures 3-(a, b)). Then, we transform each visible surface point $\mathbf{p} \in \mathsf{G}$ to the light source viewpoint (Lines 7-8 of Algorithm 2) and use H to identify potential fragments in penumbra (8) (Line 9 of Algorithm 2). The visibility of the surface points outside the penumbra is given by the shadow test (Lines 10-11 of Algorithm 2). Meanwhile, for surface points in the penumbra, we estimate the average blocker depth (6), the penumbra width (7), and perform the soft shadow filtering over revectorized hard shadows (Lines 13-15 of Algorithm 2, Figures 3-(c, d, e)).

In a naïve implementation of the relative position estimation $\mathbf{r}$ (Figure 2), one would need to traverse S several times for every sample inside w_k. To optimize RBSM, once we have computed the signed distances d_l, d_r, d_t and d_b for one fragment, we may reuse such values to define the anti-aliased visibility of neighbor shadow map samples. This prevents the algorithm from a new traversal over the shadow map, saving shadow map texture lookups. The steps of our optimized filtering are given in Algorithm 3 and in Figure 2.[‡]

[‡] The function to compute signed distances (Line 19 of Algorithm 3) is further described in the Section 1 of the supplementary document.

Given $w_k \times w_k$ samples for soft shadow filtering (Line 2 of Algorithm 3), we uniformly sample serially a shadow map texel $\tilde{\mathbf{s}}$, neighbour of $\tilde{\mathbf{p}}$ in w_p, and traverse the kernel in a row-wise order (Lines 5-7 of Algorithm 3). For each sample, we perform the hard shadow revectorization (Lines 8, 9, 21, 22 of Algorithm 3, Figures 2-(a, b, c)) and average the results over the kernel (Lines 23 and 29 of Algorithm 3). If the next sample inside the soft shadow kernel is located in the same horizontal shadow silhouette of the previous sample (Line 10 of Algorithm 3 and Figure 2-(d)), we update the previously computed signed distances for both left (d_l) and right (d_r) directions by a sum or subtraction over the distance between the two samples (Figure 2-(e)). In our case, this distance is given by the ratio between w_p and w_k (Lines 11-17 of Algorithm 3). The updated signed distances are used to estimate the relative position of the current sample in the shadow silhouette (Line 21 of Algorithm 3 and Figure 2-(f)) and perform the revectorization (Line 22 of Algorithm 3) with less shadow map texture lookups. To detect whether the current sample is located in the same silhouette of the previous sample (function **sameDirection** of Algorithm 3 and Figure 2-(d)), we check whether both samples share, at least, a single silhouette direction in the vertical direction (*i.e.*, if both samples have the third or fourth coordinate of **D** equals to 1).

RBSSM is able to generate anti-aliased soft shadows in real time. An alternative to improve its rendering performance, at the cost of some reduction in its visual quality due to shadow overestimation, relies on the usage of screen-space soft shadow filtering.

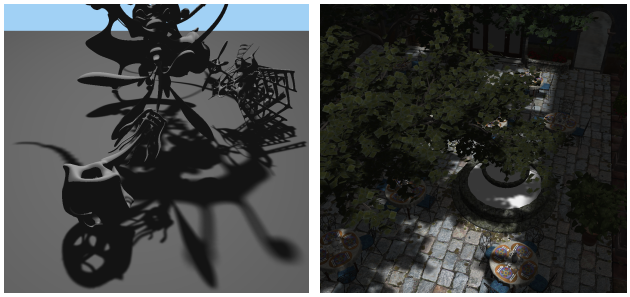
5. Screen-Space Revectorization-Based Soft Shadow Mapping

SSRBSSM is an alternative approach to compute soft shadows on the basis of RBSM and PCSS frameworks that favors rendering performance rather than visual quality. To do so, SSRBSSM computes a shadow map and a G-buffer (Figure 3-(a)). Then, the average blocker depth and penumbra width are estimated in the shadow map space as well (Figures 3-(c, d)). However, the penumbra width estimated by RBSSM works well for filtering in the shadow map space. To produce soft shadows in the screen space, we estimate a screen-space penumbra width w_p^s [MKHS10]

$$w_p^s = \frac{w_p z_{\text{screen}}}{z_{\text{eye}}}, \qquad (9)$$
$$z_{\text{screen}} = \frac{1}{2\tan\frac{fov_y}{2}},$$

where fov_y specifies the vertical field of view angle, z_{eye} is the distance of the fragment to the camera and z_{screen} is the inverse of the viewport scale, in terms of field of view.

To produce anti-aliased screen-space soft shadows, we must obtain the information about the anti-aliased hard shadows in the screen space. To do so, we produce filtered hard shadows in screen-space using the filtered visibility function of RBSM (see Figure 1-(f)) and save the filtered hard shadow intensity of each visible fragment. Next, we separate the soft shadow filtering in two steps: horizontal and vertical filtering in screen space. In the horizontal pass, filtering is performed over the filtered hard shadow on the screen-sized penumbra area. In the vertical pass, filtering is done over the horizontally filtered hard shadows. In this step, the filtering must be edge-aware, since we lose information about the edge location



(a) YeahRight (b) San Miguel

Figure 4: *Two scenarios with accurate shadows computed using the average of* 1024 *area light source samples.*

in screen space. Hence, we use the separable cross-bilateral filter technique proposed in [PvV05] for this purpose. Because the bilateral filter is not separable in essence, striped artifacts may appear during the filtering. We reduce such artifacts by increasing the bilateral filter sampling rate. Even in this case, since the number of increased samples is still smaller than the number of samples required by a non-separable implementation of the bilateral filter, the separable bilateral filter is better suited, in terms of performance, to our solution than the non-separable version of the bilateral filter.

As we show in the next section, the computation of the screen-space soft shadow filtering together with the revectorization effect makes SSRBSSM an efficient alternative for high-performance anti-aliased soft shadow rendering.

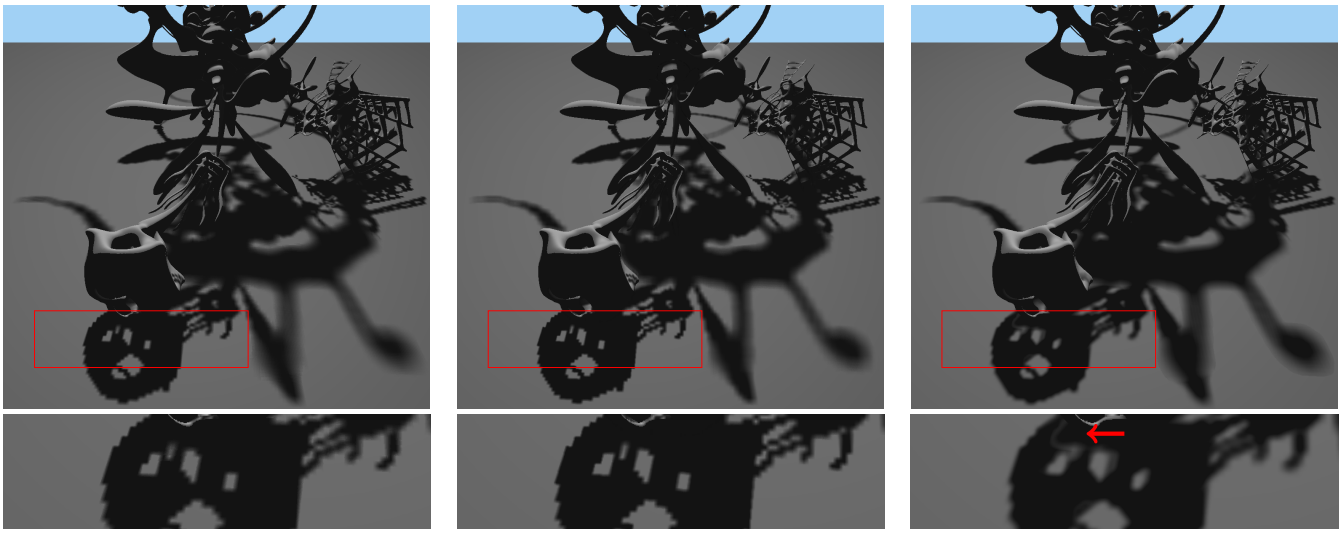
6. Results

In this section, we evaluate the techniques in terms of visual quality and performance. We compare the proposed approaches, RBSSM and SSRBSSM, with PCSS [Fer05], the most traditional real-time soft shadow technique, MSSM [PMWK16], the most recent shadow map pre-filtering technique, and SSABSS [ZS11], a screen-space soft shadow technique. Visual quality is evaluated for the same shadow map and viewport resolutions in different scenarios. Performance is evaluated for different shadow map and viewport resolutions in different scenarios as well. Hard shadow techniques are not evaluated because they are out of the scope of this paper. Temporal coherency results, as well as an extended evaluation with more techniques, are available in the supplementary video.

6.1. Experimental Setup

In our experimental setup, time usage was evaluated on an Intel Core$^{\text{TM}}$ i7-3770K CPU (3.50 GHz), 8GB RAM, and an NVIDIA GeForce GTX Titan X.

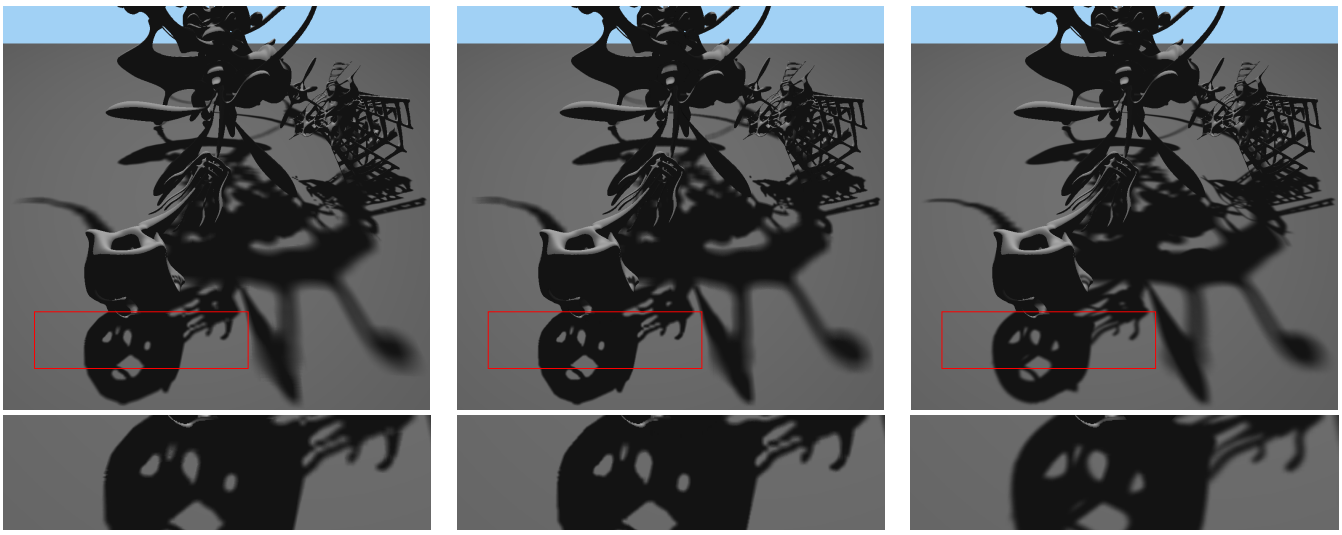
Following related work [PMWK16], we used 81 samples for the blocker search step ($w_{\text{k}}^{\text{avg}} = 9$) and 225 samples for the soft shadow filtering step ($w_{\text{k}} = 15$) for PCSS and MSSM. For RBSSM, we have used 121 samples for the soft shadow filtering ($w_{\text{k}} = 11$) since the shadow revectorization approach requires less samples to effectively minimize banding artifacts [MA16]. As noted by



(a) Percentage-Closer Soft Shadows (PCSS) - 2.0 ms

(b) Screen-Space Anisotropic Blurred Soft Shadows (SSABSS) - 3.4 ms

(c) Moment Soft Shadow Mapping (MSSM) - 2.2 ms



(d) Revectorization-based Soft Shadow Mapping (RBSSM) - 14.9 ms

(e) Screen-Space Revectorization-based Soft Shadow Mapping (SSRBSSM) - 3.6 ms

(f) Ground-Truth 78.2 ms

Figure 5: *An equal low shadow map resolution comparison between soft shadow techniques. Light leaking artifact is pointed by the red arrow in (c). Images were generated for the YeahRight model using a* 1024^2 *shadow map resolution. The ground-truth image was computed using the average of* 1024 *area light source samples.*

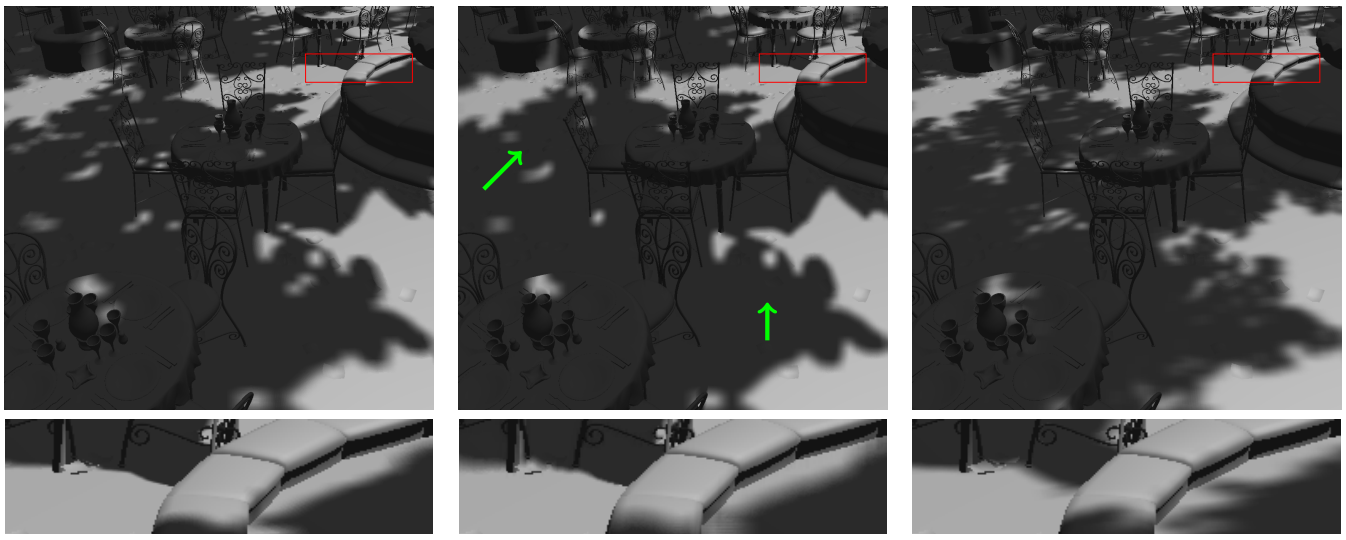
related work [BGC15], screen-space techniques typically require high-order kernel sizes to suppress banding artifacts. In this sense, we have used 529 samples ($w_k^{avg} = w_k = 23$) for both SSABSS and SSRBSSM techniques.

We have chosen two distinct scenarios to evaluate the visual quality and performance of the proposed techniques and related work. YeahRight, shown in Figure 4-(a), is a single but complex object, with fine, detailed structures along its silhouette. San Miguel, shown in Figure 4-(b), is a more complex (game-like) scenario, aimed to show how different techniques handle the soft shadow computation in a scenario with multiple objects in different scales, distributed all over the scene and overlapping each other. An extended evaluation with additional scenarios is available in both the accompanying video and Section 2 of the supplementary document.

(a) Percentage-Closer Soft Shadows (PCSS) - 5.9 ms

(b) Screen-Space Anisotropic Blurred Soft Shadows (SSABSS) - 6.5 ms

(c) Moment Soft Shadow Mapping (MSSM) - 6.0 ms



(d) Revectorization-based Soft Shadow Mapping (RBSSM) - 18.7 ms

(e) Screen-Space Revectorization-based Soft Shadow Mapping (SSRBSSM) - 6.7 ms

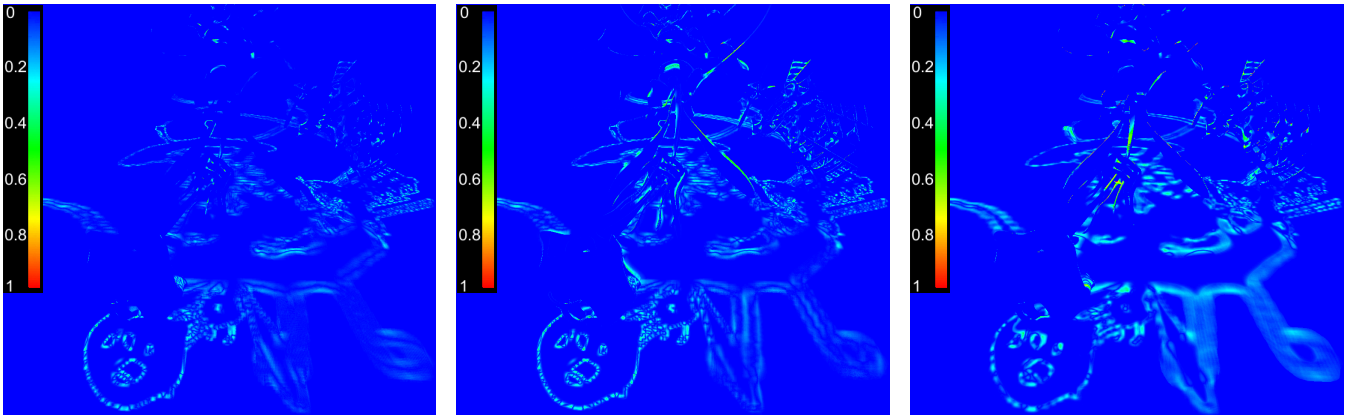
(f) Ground-Truth 1.06 s

Figure 6: *An equal low shadow map resolution comparison between soft shadow techniques. Light leaking artifact is pointed by the red arrow in (c). Shadow overestimation is pointed by the green arrow in (e). Images were generated for the San Miguel model using a* 1024^2 *shadow map resolution. The ground-truth image was computed using the average of* 1024 *area light source samples.*

6.2. Rendering Visual Quality

In this section, we provide a visual comparison between soft shadows produced under the same shadow map resolution (Figures 5 and 6), a false color visualization that measures the numerical difference of shadow intensities estimated for each technique and the ground-truth (Figures 7 and 9), and a false color visualization that uses the HDR-VDP-2 metric [MKRH11] to measure perceptual quality differences (Figures 8 and 10). Following the suggestion of the authors of the metric[§], we disabled the *spatial pooling* component and adjusted the parameter *peak sensitivity* = *3* to use this metric in our dataset. The output of this metric is the gray-scaled version of the evaluated image interpolated with a value that indicates the probability that a difference will be perceived by a human.

§ http://hdrvdp.sourceforge.net/wiki/index.php/Frequently_Asked_Questions



(a) Percentage-Closer Soft Shadows (PCSS) - RMSE: 0.0289

(b) Screen-Space Anisotropic Blurred Soft Shadows (SSABSS) - RMSE: 0.0438

(c) Moment Soft Shadow Mapping (MSSM) - RMSE: 0.0535

(d) Revectorization-based Soft Shadow Mapping (RBSSM) - RMSE: 0.0273

(e) Screen-Space Revectorization-based Soft Shadow Mapping (SSRBSSM) - RMSE: 0.0366

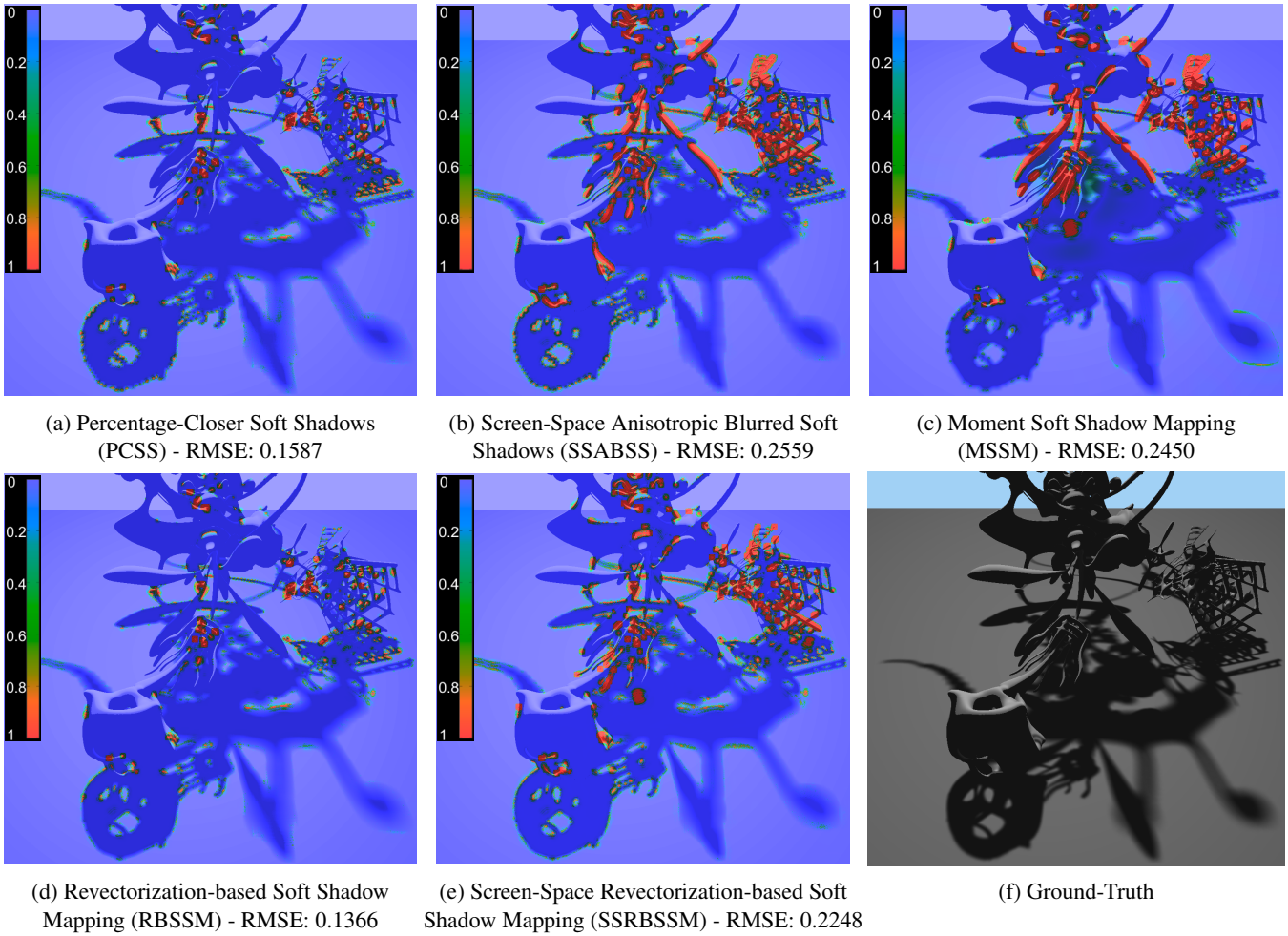
(f) Ground-Truth

Figure 7: *False color visualizations show the difference between the shadows produced by different soft shadow techniques (a, b, c, d, e) and the ground-truth one (f). Images were generated for the YeahRight model using a* 1024^2 *shadow map resolution and evaluated using the root mean squared error (RMSE). The ground-truth image was computed using the average of* 1024 *area light source samples.*

As shown in Figures 5-(a, b) and 6-(a, b), for a limited 1024^2 shadow map resolution, both PCSS and SSABSS produce aliasing artifacts along the silhouette of the penumbra. These artifacts are more visible for small penumbra sizes, as shown in the close-ups of Figures 5-(a, b) and 6-(a, b). MSSM generates small light leaking artifacts in the shadow, as pointed by the red arrows in Figures 5-(c) and 6-(c). As shown in Figures 5-(d, e) and 6-(d, e), the revectorization-based soft shadow techniques proposed in this paper are able to suppress aliasing artifacts along the penumbra, while generating soft shadows that resemble the ones obtained on the ground-truth, as mainly visible in Figure 5-(f). Moreover, the proposed techniques produce plausible soft shadows for static and dynamic shadows projected on planar and non-planar receivers, supporting self-shadowing and the computation of the shadows projected by overlapping distant objects.

All the techniques evaluated in this section are not able to simulate accurately the size and the quality of the penumbra effect obtained by ground-truth shadows, as mainly visible in Figure 6-(f). Also, SSRBSSM is more prone to shadow overestimation than RBSSM (as pointed by the green arrows in Figure 6-(e)). That happens because SSRBSSM makes use of the filtered hard shadow visibility function (5) of RBSM (Figure 1-(f)), whose smoothness is essential for the separable bilateral filtering, but since the revectorization extends the penumbra into lit regions, this visibility function is prone to hole filling for small lit regions. On the other hand, RBSSM uses the hard shadow variant (4) of RBSM (Figure 1-(e)), that also extends umbra into lit regions, but is known to produce less overestimation artifacts than the filtered hard shadow variant [MA18].

For the YeahRight model (Figures 7 and 8), MSSM is not only the most numerically inaccurate technique (Figure 7-(c)), but is also the second-most perceptually inaccurate technique (Figure 8-(c)). In order to reduce the presence of light leaking artifacts in the fi-



(a) Percentage-Closer Soft Shadows (PCSS) - RMSE: 0.1587

(b) Screen-Space Anisotropic Blurred Soft Shadows (SSABSS) - RMSE: 0.2559

(c) Moment Soft Shadow Mapping (MSSM) - RMSE: 0.2450

(d) Revectorization-based Soft Shadow Mapping (RBSSM) - RMSE: 0.1366

(e) Screen-Space Revectorization-based Soft Shadow Mapping (SSRBSSM) - RMSE: 0.2248

(f) Ground-Truth

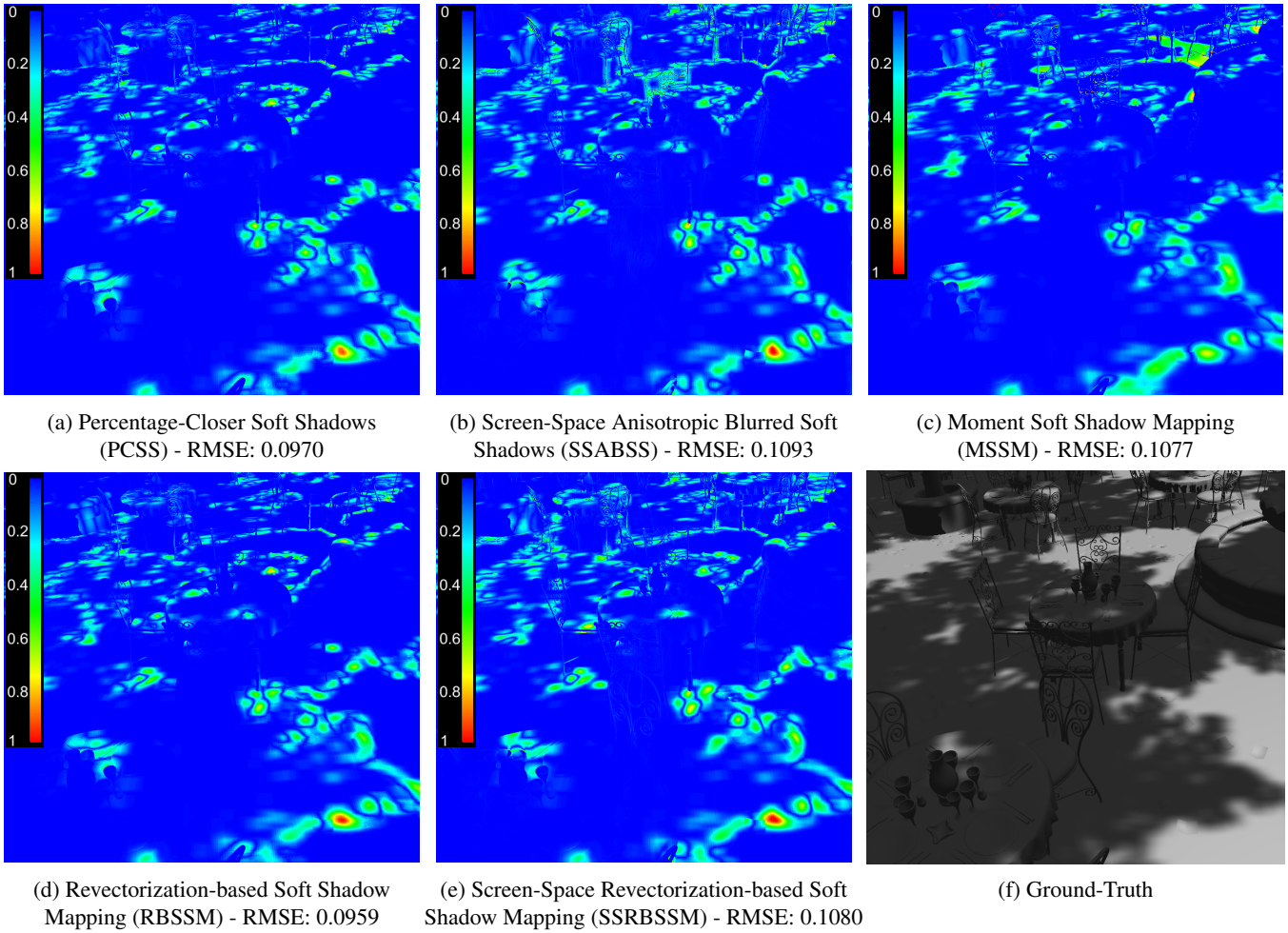
Figure 8: *HDR-VDP-2 metric shows the perceptual difference between the shadows produced by different soft shadow techniques (a, b, c, d, e) and the ground-truth one (f). Images were generated for the YeahRight model using a* 1024^2 *shadow map resolution. In the sub-captions, we show the root mean squared error (RMSE) values measured from the perceptual metric. The ground-truth image was computed using the average of* 1024 *area light source samples.*

nal image, MSSM overestimates the shadow near the penumbra region, causing the generation of numerical and perceptual errors. As shown in Figures 7-(b) and 8-(b), even by operating in screen space, SSABSS is able to generate soft shadows with similar accuracy than the ones generated by MSSM. However, aliasing and light leaking artifacts are strongly perceived by the user in the final image (see the red pixels in Figure 8-(b)). At a slightly increased processing time, SSRBSSM is able to reduce most of the aliasing and light leaking artifacts generated by SSABSS, as mainly visible in Figure 8-(e). However, as a screen-space real-time approach, the technique still generates shadows that deviate from the ground-truth one. PCSS (Figures 7-(a) and 8-(a)) generates shadows typically more accurate than pre-filtering (MSSM) and screen-space (SSABSS and SSRBSSM) approaches, but suffers from the presence of aliasing artifacts along the penumbra, more visible in the closeup of Figure 5-(a). In this sense, RBSSM reduces those aliasing artifacts (Figures 7-(d) and 8-(d)), generating the most numerical and perceptual accurate soft shadows between the real-time soft shadow techniques.

For the complex San Miguel scenario (Figures 9 and 10), none of the real-time soft shadow techniques are able to simulate the penumbra effect accurately, proving that these techniques are more suitable for simpler models, such as the YeahRight (Figure 5), and that the problem of real-time, accurate penumbra simulation is still unsolved for complex scenarios.

6.3. Rendering Performance

In Figures 11 and 12, we show the performance of the soft shadow techniques evaluated in this section under varying shadow map and output resolutions for the different scenarios shown in Figure 4.



(a) Percentage-Closer Soft Shadows (PCSS) - RMSE: 0.0970

(b) Screen-Space Anisotropic Blurred Soft Shadows (SSABSS) - RMSE: 0.1093

(c) Moment Soft Shadow Mapping (MSSM) - RMSE: 0.1077

(d) Revectorization-based Soft Shadow Mapping (RBSSM) - RMSE: 0.0959

(e) Screen-Space Revectorization-based Soft Shadow Mapping (SSRBSSM) - RMSE: 0.1080

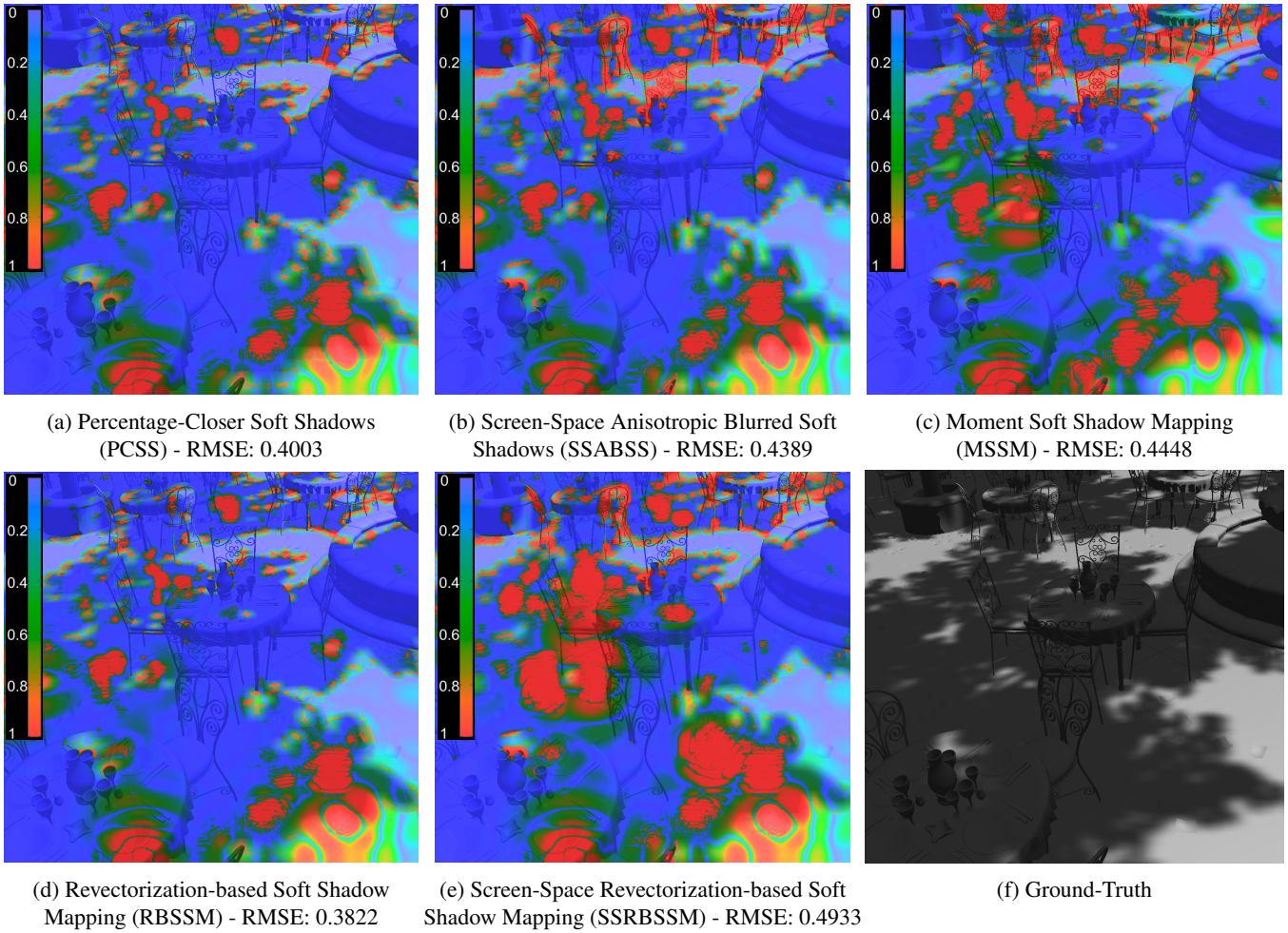
(f) Ground-Truth

Figure 9: *False color visualizations show the difference between the shadows produced by different soft shadow techniques (a, b, c, d, e) and the ground-truth one (f). Images were generated for the San Miguel model using a* 1024^2 *shadow map resolution and evaluated using the root mean squared error (RMSE). The ground-truth image was computed using the average of* 1024 *area light source samples.*

PCSS and MSSM are generally the fastest techniques, regardless of the scene configuration. The screen-space techniques (SSABSS and our SSRBSSM) are slightly slower than PCSS and MSSM, providing competitive performance for varying shadow map resolution (Figure 11), but being not that scalable with respect to the output resolution, as shown in Figure 12. In this sense, it is noteworthy that our SSRBSSM technique is only $\sim$ 0.1 ms ($\sim$ 1%) slower than SSABSS and only $\sim$ 1.6 ms ($\sim$ 23%) slower than the PCSS and MSSM techniques, meanwhile our technique provides improved visual quality by the reduction of the aliasing artifacts commonly generated by these techniques. RBSSM has the same performance characteristics of related work, but the additional cost demanded by the light-space revectorization-based filtering makes RBSSM slower than related work.

In Figures 11 and 12, we also provide a comparison between our optimized and unoptimized implementations of RBSSM (Section 4). By using the strategy defined in Algorithm 3 and the penumbra classification provided by HSM, we are able to reduce processing time by $\sim$ 1.7 ms ($\sim$ 7.6%) when compared to a naïve implementation of the RBSSM algorithm. 29.4 % ($\sim$ 0.5 ms) of this reduced processing time is obtained by the HSM optimization and the other 70.6 % ($\sim$ 1.2 ms) is provided by the optimization of Algorithm 3. Considering that the HSM is orthogonal to our solution and could be adapted to optimize related work, slowing down our approach by 0.5 ms enables a more accurate numerical comparison between related work and RBSSM without using HSM.

In Figure 13, we show how much each step of both revectorization-based techniques contributes to the total frame time obtained for each scenario. Although G-Buffer rendering, shadow map rendering and Phong shading steps require the same amount of time for both RBSSM and SSRBSSM, these steps consume more of the total processing time for the SSRBSSM technique, since



(a) Percentage-Closer Soft Shadows (PCSS) - RMSE: 0.4003

(b) Screen-Space Anisotropic Blurred Soft Shadows (SSABSS) - RMSE: 0.4389

(c) Moment Soft Shadow Mapping (MSSM) - RMSE: 0.4448

(d) Revectorization-based Soft Shadow Mapping (RBSSM) - RMSE: 0.3822

(e) Screen-Space Revectorization-based Soft Shadow Mapping (SSRBSSM) - RMSE: 0.4933

(f) Ground-Truth

Figure 10: *HDR-VDP-2 metric shows the perceptual difference between the shadows produced by different soft shadow techniques (a, b, c, d, e) and the ground-truth one (f). Images were generated for the San Miguel model using a* 1024^2 *shadow map resolution. In the sub-captions, we show the root mean squared error (RMSE) values measured from the perceptual metric. The ground-truth image was computed using the average of* 1024 *area light source samples.*

this technique is faster than RBSSM. RBSM (Section 3) typically consumes less than 0.5 milliseconds for anti-aliased filtered hard shadow rendering, or, alternatively, less than 10% of the total processing time (see the red HS bar in Figure 13), proving to be an efficient alternative to shadow mapping. As for RBSSM, the hierarchical shadow map is built in less than 1 millisecond (or 5% of the RBSSM total frame time), not introducing a significant overhead for the technique (see the blue HS bar in Figure 13). Since screen-space techniques require high-order kernel sizes for both average blocker depth and soft shadow filtering [BGC15], the average blocker depth estimation is slower for SSRBSSM when compared to RBSSM. Finally, the bottleneck of the RBSSM technique lies in the soft shadow filtering step, which requires more than 80% of the frame time. By performing the filtering step in screen space, we are able to reduce more than 90% of the processing time demanded by the filtering step of RBSSM, keeping the technique competitive with respect to the performance obtained by related work, at the cost of slightly decreased visual quality.

6.4. Discussion

Comparing solely the results obtained with RBSSM and SSRBSSM, the screen-space variant is able to compute soft shadows visually similar to the ones obtained with RBSSM, as can be seen in Figure 5 and in the Section 2 of the supplementary document, but this technique produces more shadow overestimation than RBSSM, as pointed by the green arrows in Figure 6 and as shown in Figure 10-(e), due to the usage of the filtered hard shadow visibility. In this sense, the greatest advantage of SSRBSSM is that, by performing most of the computation in the screen space, the technique is about 2 to 4 times faster than RBSSM for the tested scenarios. However, by performing the soft shadow filtering in light

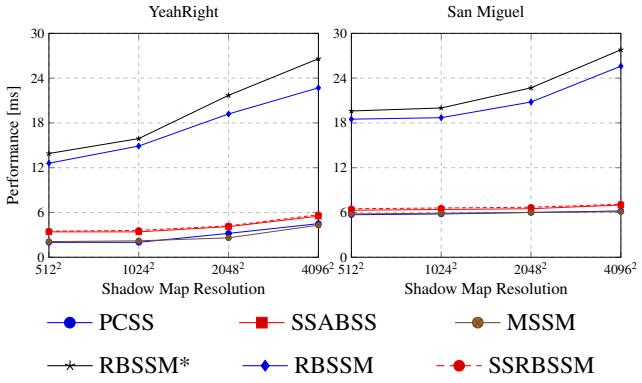


Figure 11: *Time usage (in milliseconds) for several soft shadow techniques and scenes shown in Figures 5 and 6 at an output* 1280×720 *resolution. Measurements include varying shadow map resolution. RBSSM* refers to the unoptimized implementation of RBSSM.*

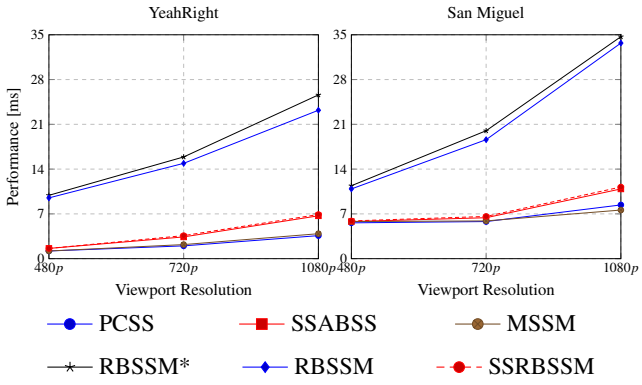


Figure 12: *Time usage (in milliseconds) for several soft shadow techniques and scenes shown in Figures 5 and 6 at an* 1024^2 *shadow map resolution. Measurements include varying output resolution. RBSSM* refers to the unoptimized implementation of RBSSM.*

space, RBSSM has higher accuracy than SSRBSSM, as measured in Figures 7, 8, 9, and 10. Thus, RBSSM is suitable for applications that demand **high-quality** shadow rendering, meanwhile SSRBSSM is desirable for applications that demand visually plausible soft shadow rendering with **high performance**.

Compared to related work, RBSSM and SSRBSSM are able to reduce aliasing and light leaking artifacts even for low-resolution shadow maps (Figures 5-(d, e) and 6-(d, e)). Also, RBSSM is the most accurate technique with respect to the HDR-VDP-2 metric (Figures 8-(d) and 10-(d)), although shadow details can get lost due to the shadow revectorization (see the closeup of Figure 6-(d)). In terms of performance, as can be seen in Figures 11 and 12, all the techniques exhibit a similar behaviour with respect to the variation of the shadow map resolution. However, screen-space techniques tend to be less scalable than light-space techniques for high viewport resolutions. Nevertheless, most of the techniques (namely PCSS, MSSM and SSABSS) achieve low running times by generating shadows prone to aliasing and light leaking artifacts. In this sense, SSRBSSM is an exception, since the technique uses shadow revectorization to minimize the aliasing. Hence, we can state that SSRBSSM is desirable for simple (Figure 5) and complex scenarios (Figure 6), since the technique is less prone to aliasing and light leaking artifacts than related work and provides one of the fastest processing times, even if it is still susceptible to shadow overestimation artifacts. Meanwhile, RBSSM is desirable for scenarios that demand improved visual quality at the cost of a lower frame rate.

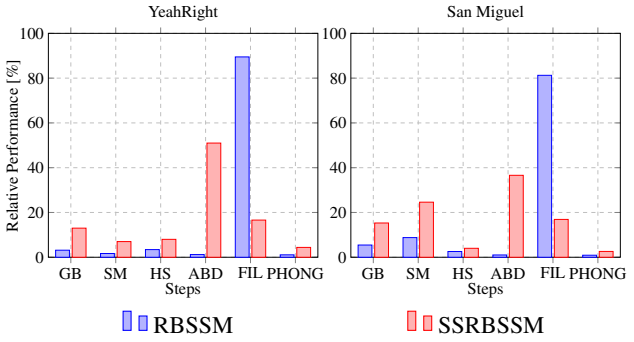
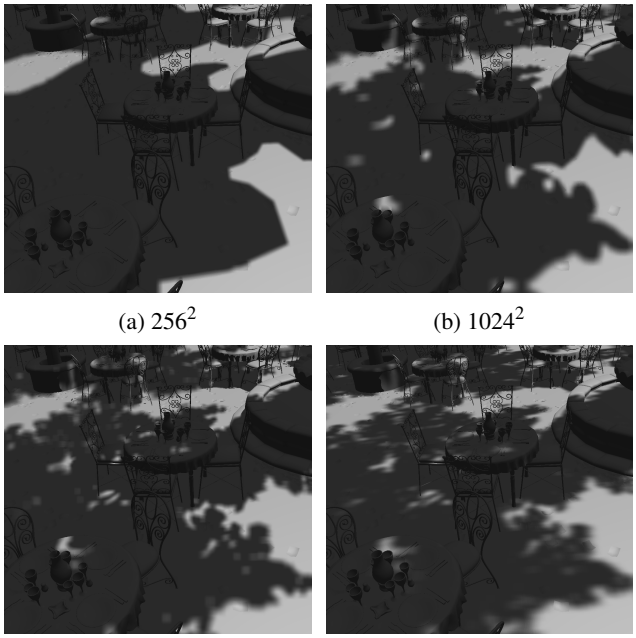


Figure 13: *Time usage (in %) for the main steps of both RBSSM and SSRBSSM techniques in the scenes shown in Figures 5 and 6. Steps: GB - G-Buffer rendering. SM - Shadow map rendering. HS - HSM building (for RBSSM) or filtered Hard Shadow rendering (for SSRBSSM). ABD - Average blocker depth computation. FIL - Soft shadow filtering. PHONG - Phong shading. Steps not covered in this figure (e.g., penumbra size estimation) required less than* 1% *of the total frame time obtained by our experimental setup.*

We believe that the proposed approaches can be integrated with other strategies that are well known to improve the shadow map resolution and reduce the shadow aliasing. For instance, the use of partitioning techniques [LSL11], or the use of compressed high-resolution shadow maps [SBE16], are mainly useful when rendering shadows in large outdoor scenarios. In this sense, to produce revectorization-based soft shadows with these techniques, one just needs to evaluate the proposed visibility functions over the partitioned/compressed shadow maps.

As can be seen in the closeups shown in Figure 5, shadow aliasing artifacts are more prominent in small penumbra sizes rather than large penumbra sizes. Larger light sources tend to produce larger penumbra sizes, helping in the hiding of aliasing artifacts. However, in this case, higher kernel sizes must be used to avoid banding artifacts in the soft shadow rendering. Hence, RBSSM and SSRBSSM are more useful to suppress aliasing artifacts present in near contact shadows or small penumbra regions, scenarios where the aliased shadows are more apparent. In this sense, one must be aware that RBSSM provides higher visual quality at the cost of lower performance, meanwhile SSRBSSM runs two to four times faster than RBSSM, while introducing shadow overestimation artifacts in the final rendering.



(a) 256^2 (b) 1024^2

(c) 4096^2 (d) Ground-Truth

Figure 14: *SSRBSSM is unable to capture fine details along the shadow silhouette and suffers from shadow overestimation, mainly for low-resolution shadow maps (a, b). Those problems are effectively reduced by increasing the shadow map resolution (c). Images were generated for the San Miguel model.*

6.5. Limitations

RBSSM, SSRBSSM, as well as the other real-time techniques compared in Section 6.2, compute soft shadows using only one shadow map, located at the center of the area light source, and assuming that both light source and blocker objects are planar and parallel to each other. This simplifying assumption makes the algorithms fast enough to run in real time and to generate visually plausible soft shadows. However, physical accuracy is lost because that assumption leads to an incorrect penumbra size (see the false color visualizations of Figures 9 and 10).

Despite the real-time performance provided by SSRBSSM, the technique suffers from the same problem of the other screen-space soft shadow techniques, such as SSABSS: the screen-space filtering does not approximate the filtering produced from the perspectively deformed kernels used for soft shadow filtering in light space, nor takes into account the shadows located outside the view. Moreover, SSRBSSM and MSSM share the same drawback of shadow overestimation for some complex scenarios, such as San Miguel (Figure 6). In this sense, the shadows produced by SSRBSSM are less accurate than the ones produced by techniques that perform the filtering in light space. However, as we show in Figures 5-(e) and in the Section 2 of the supplementary document, SSRBSSM generally produces soft shadows with less aliasing and light leaking artifacts than related work for scenarios with near contact shadows or small penumbra sizes.

RBSSM and SSRBSSM are similar to the other soft shadow mapping techniques in the sense that the quality of the rendered soft shadows is dependent on the resolution of the shadow map used. As shown in Figure 14-(a), for instance, for a 256^2 shadow map resolution, SSRBSSM generates anti-aliased soft shadows with a high presence of shadow overestimation caused by the own visibility function of the filtering variant of RBSM (Figure 1-(f)) that puts in umbra small lit regions that are surrounded by shadow [MA16], but also caused by the limited resolution of the shadow map, that does not capture accurately the depth of the light blocker objects. As shown in Figure 14-(b, c) and in Figure 11 of the supplementary document, as long as the shadow map resolution is increased, the visual quality of all soft shadow mapping techniques is increased as well, being closer to the ground-truth (Figure 14-(d)).

RBSSM is slower than related work for the same shadow map and viewport resolutions, because the technique uses a non-separable filter that performs filtering in light space on the basis of revectorized shadows. As already discussed in Section 6.4, this technique still provides real-time performance and produces shadows that outperform related work in terms of visual quality.

7. Conclusion and Future Work

In this paper, we have proposed the use of revectorization to reduce aliasing and light leaking artifacts found in soft shadows. The RBSSM technique extends the PCSS framework to take into account the revectorization effect produced by a revectorization-based filtering algorithm. With this approach, we increased the processing time of the soft shadow rendering to minimize the shadow aliasing problem. A screen-space variant of RBSSM, the SSRBSSM, was proposed to bring higher performance to the revectorization approach, while generating soft shadows with less aliasing and light leaking artifacts than related work, but still suffering from shadow overestimation. We have shown that our techniques are real time and produce visually plausible soft shadows mainly for scenarios with near contact shadows or small penumbra sizes produced by low-resolution shadow maps.

In future work, temporal coherence [SLSW13] could be exploited to reuse some soft shadow calculations, further improving the performance of the revectorization-based soft shadow techniques. Another possibility for future work is the proposition of a hybrid approach that uses RBSSM whenever visual quality is the priority for the scene rendering, and SSRBSSM whenever performance is the priority or RBSSM would be costly to compute the soft shadows. Finally, the integration of the proposed techniques in the context of a game engine would allow the evaluation of both revectorization-based approaches in industrial applications, such as games. This would further enable the investigation of the benefits of RBSSM and SSRBSSM in more complex scenarios.

Acknowledgments

This work was supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and by the NVIDIA Corporation, who provided the GeForce Titan X through the GPU Education Center program. YeahRight model is courtesy of Keenan Crane. San Miguel 2.0 is courtesy of Morgan McGuire [McG17].

References

[ADM*08] ANNEN T., DONG Z., MERTENS T., BEKAERT P., SEIDEL H.-P., KAUTZ J.: Real-time, all-frequency shadows in dynamic scenes. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 1–8. 2

[AMS*08] ANNEN T., MERTENS T., SEIDEL H.-P., FLERACKERS E., KAUTZ J.: Exponential shadow maps. In *Proceedings of GI* (2008), pp. 155–161. 1, 2

[BCS08] BAVOIL L., CALLAHAN S. P., SILVA C. T.: Robust soft shadow mapping with backprojection and depth peeling. *J. Graphics Tools 13*, 1 (2008), 19–30. 2

[BGC15] BUADES J. M., GUMBAU J., CHOVER M.: Separable soft shadow mapping. *The Visual Computer 32*, 2 (2015), 167–178. 2, 7, 12

[Cro84] CROW F. C.: Summed-area tables for texture mapping. In *Proceedings of the ACM SIGGRAPH* (1984), pp. 207–212. 2

[DL06] DONNELLY W., LAURITZEN A.: Variance shadow maps. In *Proceedings of the ACM I3D* (2006), pp. 161–165. 1, 2

[DY10] DONG Z., YANG B.: Variance soft shadow mapping. In *Proceedings of the ACM I3D* (2010), pp. 1–1. 2

[ESAW11] EISEMANN E., SCHWARZ M., ASSARSSON U., WIMMER M.: *Real-Time Shadows*. A.K. Peters, 2011. 1, 2, 5

[Fer05] FERNANDO R.: Percentage-closer soft shadows. In *ACM SIGGRAPH Sketches* (2005). 1, 5, 6

[GBP06] GUENNEBAUD G., BARTHE L., PAULIN M.: Real-time soft shadow mapping by backprojection. In *Proceedings of the EGSR* (2006), pp. 227–234. 2, 5

[GBP07] GUENNEBAUD G., BARTHE L., PAULIN M.: High-quality adaptive soft shadow mapping. *Computer Graphics Forum 26*, 3 (2007), 525–533. 2

[Lau08] LAURITZEN A.: Summed-area variance shadow maps. In *GPU Gems 3*, Nguyen H., (Ed.). Addison-Wesley, 2008, pp. 157–182. 2

[LM08] LAURITZEN A., MCCOOL M.: Layered variance shadow maps. In *Proceedings of GI* (2008), pp. 139–146. 1

[LMSG14] LECOCQ P., MARVIE J.-E., SOURIMANT G., GAUTRON P.: Sub-pixel Shadow Mapping. In *Proceedings of the ACM I3D* (2014), pp. 103–110. 2

[LSL11] LAURITZEN A., SALVI M., LEFOHN A.: Sample Distribution Shadow Maps. In *Proceedings of the ACM I3D* (2011), pp. 97–102. 13

[LSMD15] LIKTOR G., SPASSOV S., MÜCKL G., DACHSBACHER C.: Stochastic soft shadow mapping. In *Proceedings of the EGSR* (2015), pp. 1–11. 2

[MA16] MACEDO M., APOLINARIO A.: Revectorization-based shadow mapping. In *Proceedings of the GI* (2016), pp. 75–83. 1, 2, 6, 14

[MA17a] MACEDO M., APOLINARIO A.: Euclidean distance transform soft shadow mapping. In *Proceedings of the SIBGRAPI* (2017), pp. 238–245. 2

[MA17b] MACEDO M. C. F., APOLINARIO JR. A. L.: Revectorization-Based Accurate Soft Shadow Using Adaptive Area Light Source Sampling. In *Proceedings of the GI* (2017), pp. 181–189. 2

[MA18] MACEDO M., APOLINÁRIO A.: Improved anti-aliasing for euclidean distance transform shadow mapping. *Computers & Graphics 71* (2018), 166 – 179. 9

[McG17] MCGUIRE M.: Computer graphics archive, July 2017. `https://casual-effects.com/data`. 14

[MFB08] MYERS K., FERNANDO R., BAVOIL L.: *Integrating Realistic Soft Shadows into Your Game Engine*. Tech. rep., NVIDIA, 2008. 5

[MKHS10] MOHAMMADBAGHER M., KAUTZ J., HOLZSCHUCH N., SOLER C.: Screen-space percentage-closer soft shadows. In *ACM SIGGRAPH Posters* (2010), pp. 133–133. 2, 6

[MKRH11] MANTIUK R., KIM K. J., REMPEL A. G., HEIDRICH W.: HDR-VDP-2: A Calibrated Visual Metric for Visibility and Quality Predictions in All Luminance Conditions. *ACM Trans. Graph. 30*, 4 (July 2011), 40:1–40:14. 8

[MTAA17] MACEDO M. C. F., TEIXEIRA A. V., APOLINARIO A. L., AGUERO K. A.: Hard Shadow Anti-Aliasing for Spot Lights in a Game Engine. In *Proceedings of the SBGAMES* (2017), pp. 106–115. 2

[PK15] PETERS C., KLEIN R.: Moment shadow mapping. In *Proceedings of the ACM I3D* (2015), pp. 7–14. 1, 2

[PMWK16] PETERS C., MÜNSTERMANN C., WETZSTEIN N., KLEIN R.: Beyond hard shadows: Moment shadow maps for single scattering, soft shadows and translucent occluders. In *Proceedings of the ACM I3D* (2016), pp. 159–170. 2, 6

[PvV05] PHAM T., VAN VLIET L.: Separable bilateral filtering for fast video preprocessing. In *Proceedings of the IEEE ICME* (2005). 2, 6

[RSC87] REEVES W. T., SALESIN D. H., COOK R. L.: Rendering antialiased shadows with depth maps. In *Proceedings of the ACM SIGGRAPH* (1987), pp. 283–291. 1, 2, 5

[Sal08] SALVI M.: Rendering filtered shadows with exponential shadow maps. In *ShaderX 6.0 Advanced Rendering Techniques*. Charles River Media, 2008, pp. 257–274. 1, 2

[SBE16] SCANDOLO L., BAUSZAT P., EISEMANN E.: Merged Multiresolution Hierarchies for Shadow Map Compression. *Computer Graphics Forum 35*, 7 (2016), 383–390. 13

[SDMS15] SELGRAD K., DACHSBACHER C., MEYER Q., STAMMINGER M.: Filtering multi-layer shadow maps for accurate soft shadows. *Computer Graphics Forum 34*, 1 (2015), 205–215. 2

[SFY13] SHEN L., FENG J., YANG B.: Exponential soft shadow mapping. *Computer Graphics Forum 32*, 4 (2013), 107–116. 2

[SLSW13] SCHWÄRZLER M., LUKSCH C., SCHERZER D., WIMMER M.: Fast percentage closer soft shadows using temporal coherence. In *Proceedings of the ACM I3D* (2013), pp. 79–86. 14

[SS98] SOLER C., SILLION F. X.: Fast calculation of soft shadow textures using convolution. In *Proceedings of the ACM SIGGRAPH* (1998), pp. 321–332. 2

[SS07] SCHWARZ M., STAMMINGER M.: Bitmask soft shadows. *Computer Graphics Forum 26*, 3 (2007), 515–524. 2

[ST90] SAITO T., TAKAHASHI T.: Comprehensible rendering of 3-d shapes. In *Proceedings of ACM SIGGRAPH* (1990), pp. 197–206. 5

[WHL15] WYMAN C., HOETZLEIN R., LEFOHN A.: Frustum-traced Raster Shadows: Revisiting Irregular Z-buffers. In *Proceedings of the ACM I3D* (2015), pp. 15–23. 2

[Wil78] WILLIAMS L.: Casting curved shadows on curved surfaces. In *Proceedings of the ACM SIGGRAPH* (1978), pp. 270–274. 1, 3

[WP12] WOO A., POULIN P.: *Shadow Algorithms Data Miner*. CRC Press, 2012. 2

[XTP07] XIE F., TABELLION E., PEARCE A.: Soft shadows by ray tracing multilayer transparent shadow maps. In *Proceedings of the EGSR* (2007), pp. 265–276. 2

[YDF*10] YANG B., DONG Z., FENG J., SEIDEL H.-P., KAUTZ J.: Variance soft shadow mapping. *Computer Graphics Forum 29*, 7 (2010), 2127–2134. 2

[YFGL09] YANG B., FENG J., GUENNEBAUD G., LIU X.: Packet-based hierarchal soft shadow mapping. In *Proceedings of the EGSR* (2009), pp. 1121–1130. 2

[ZS11] ZHENG Z., SAITO S.: Screen space anisotropic blurred soft shadows. In *ACM SIGGRAPH Posters* (2011), pp. 75–75. 2, 6