Rectilinear Texture Warping for Fast Adaptive Shadow Mapping

Paul Rosen*
Unversity of Utah

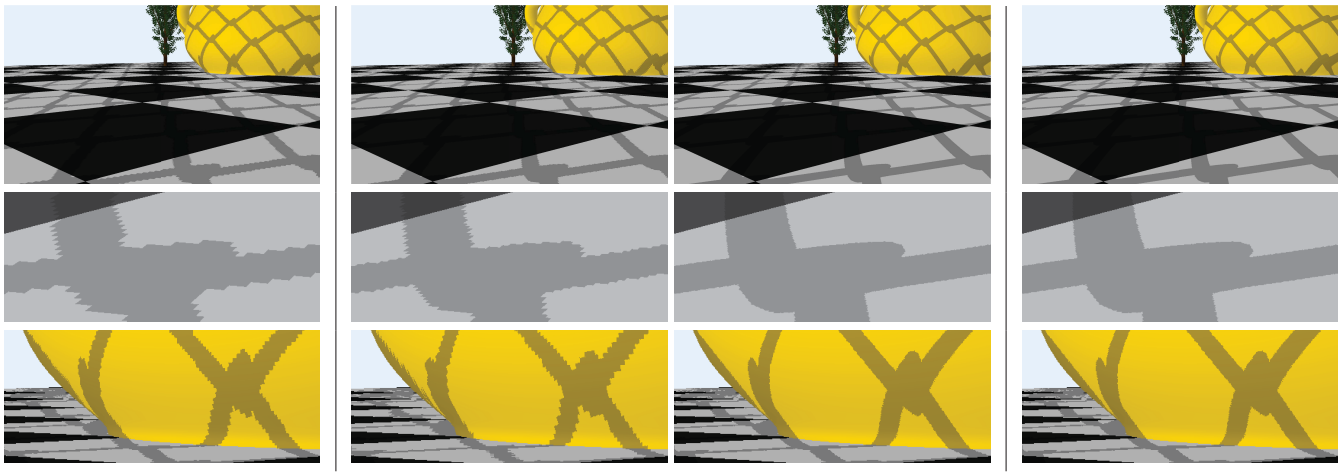


Figure 1: *Column 1: A conventional shadow map at 2,048×2,048 used to render the scene at 1080p (row 1) along with 2 zoomed regions (rows 2 and 3) in 3.3 ms. Column 2: The Rectilinear Texture Warping (RTW) shadow map at 512×512 (4.2 ms, tessellation disabled, maximum effective resolution (MER) 2,861×4,083) produces similar results to the larger conventional shadow map. Column 3: The RTW shadow map at 2,048×2,048 (5.2 ms, tessellation disabled, MER 11,448×16,333) approaches the quality of a raytraced shadow (column 4). The shadow maps used are shown in Figure 2.*

Abstract

Conventional shadow mapping relies on uniform sampling for producing hard shadow in an efficient manner. This approach trades image quality in favor of efficiency. A number of approaches improve upon shadow mapping by combining multiple shadow maps or using complex data structures to produce shadow maps with multiple resolutions. By sacrificing some performance, these adaptive methods produce shadows that closely match ground truth.

This paper introduces Rectilinear Texture Warping (RTW) for efficiently generating adaptive shadow maps. RTW images combine the advantages of conventional shadow mapping - a single shadow map, quick construction, and constant time pixel shadow tests, with the primary advantage of adaptive techniques - shadow map resolutions which more closely match those requested by output images. RTW images consist of a conventional texture paired with two 1-D warping maps that form a rectilinear grid defining the variation in sampling rate. The quality of shadows produced with RTW shadow maps of standard resolutions, i.e. 2,048×2,048 texture for 1080p output images, approaches that of raytraced results while low overhead permits rendering at hundreds of frames per second.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—[Visible Surface Algorithms, Shadowing]

Keywords: Rendering, Shadow Algorithms, Adaptive Sampling

*e-mail: prosen@sci.utah.edu

1 Introduction

Rendering visually compelling images at interactive rates continues to be an important challenge in computer graphics requiring balance between computational efficiency and retaining the highest possible image quality. One simple, efficient, and powerful approach used to improving image quality in interactive applications is shadow mapping [Williams 1978]. Shadow mapping renders a depth image of the scene from a light's perspective and uses that data to perform a constant time shadow test on output pixels. When measured against more accurate methods, such as raytracing [Whitted 1980], shadow mapping produces less accurate shadows with significantly lower computational cost.

The conventional approach to shadow mapping is to capture the shadow data on a uniform grid, independent of scene and viewing conditions. This restricts the output image quality in many situations. For example, the shadow edges, regions where illumination transitions from in-light to in-shadow, should have the same smooth shape as the silhouette of the shadow casting object. A blocky appearance is a sign frequently associated with inadequate shadow map resolution (Figure 1, column 1). In reality, various regions of the scene carry very different importance to the output image quality, a fact ignored by conventional shadow mapping. A better solution is to produce Adaptive Shadow Maps (ASMs) [Fernando et al. 2001] that dynamically allocate shadow map samples to regions with greater impact to achieving the highest possible shadow quality given limited texture and computational resources. Unfortunately, ASMs remain inefficient to produce and use.

This paper introduces a new approach to Adaptive Shadow Mapping by using Rectilinear Texture Warping (RTW) to produce shadow maps that adapt to view, scene, and lighting conditions. RTW produces single-layer textures with importance-based sampling that fit efficiently into a conventional texture. RTW images are composed of a rectilinear warping map on top of a conventional texture. The warping map enables variation of the sampling rate while the conventional texture enables high throughput performance. The quality of shadows rendered using RTW is substantially better than conven-

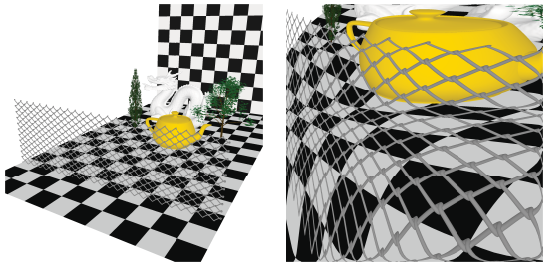


Figure 2: *The shadow maps used in Figure 1 for conventional shadow map (left) and RTW shadow mapping (right).*

tional shadows maps of the same resolution, while the expense of generating RTW shadow maps over conventional ones is small.

Figure 1 shows a sample scene composed of objects with various levels of shadow complexity. While the construction of the conventional shadow map (Figure 2, left) remains static, the RTW shadow map (Figure 2, right) dynamically adapts to changes in the light position, scene geometry, and user's desired viewpoint to produce high quality shadows given limited texture resolution. For the viewpoint within Figure 1, the conventional shadow map (column 1), even at 2,048×2,048 resolution, shows significant aliasing along the shadow edges. Using an RTW shadow map at only 512×512 resolution (column 2) shows approximately the same level of aliasing artifacts while the RTW at 2,048×2,048 (column 3) shows results of superior quality approaching that of the raytracing result (column 4). The resolution needed to create a conventional shadow map matching the maximum sampling rate of the RTW texture, referred to as the *maximum effective resolution (MER)*, is 11,448×16,333 for the 2,048×2,048 RTW shadow map. Storing a texture of that size would require over 700 MB as compared to just over 16 MB for the RTW shadow map, using 32-bit shadow map buffers. The computational cost of using an RTW shadow map is slightly above that of conventional shadow mapping. The view in Figure 1 took 5.2 ms to render with RTW as compared to 3.3 ms with conventional shadow mapping. The reader is also referred to the accompanying video for additional examples.

2 Previous Work

Shadow casting is a well-studied problem within computer graphics. As such, we focuses on methods for creating hard shadows, in particular shadow mapping. While techniques such as shadow volumes [Crow 1977] and raytracing [Whitted 1980] can also be used for generating hard shadows, shadow mapping remains most relevant to our approach.

The method most commonly used for calculating shadows in interactive applications is shadow mapping [Williams 1978; Woo 1992; Brabec et al. 2000; Pagot et al. 2004]. Conventional shadow mapping is an easy to implement method with low computational cost, good for calculating the shadows of complex objects. Unfortunately, the uniform sampling of conventional shadow mapping causes severe aliasing effects in regions with inadequate resolution (Figure 1, left). A number of adaptive, variable resolution, and multiscale methods have been developed to address the aliasing limitations of conventional shadow maps.

The irregular z-buffer [Aila and Laine 2004; Johnson et al. 2005] samples the shadow map only at locations which are visible in the output view. The resulting shadow is a perfect hard shadow which is efficiently rendered in a SIMD environment using irregular data structures. In this case, using irregular data structures works well when the shadow points are well distributed. However, shadow maps can have hot spots where many samples gather, severely affecting performance. Other advances [Sintorn et al. 2008] have improved these irregular techniques reducing the effects of sampling hot spots, though not completely eliminating the problem.

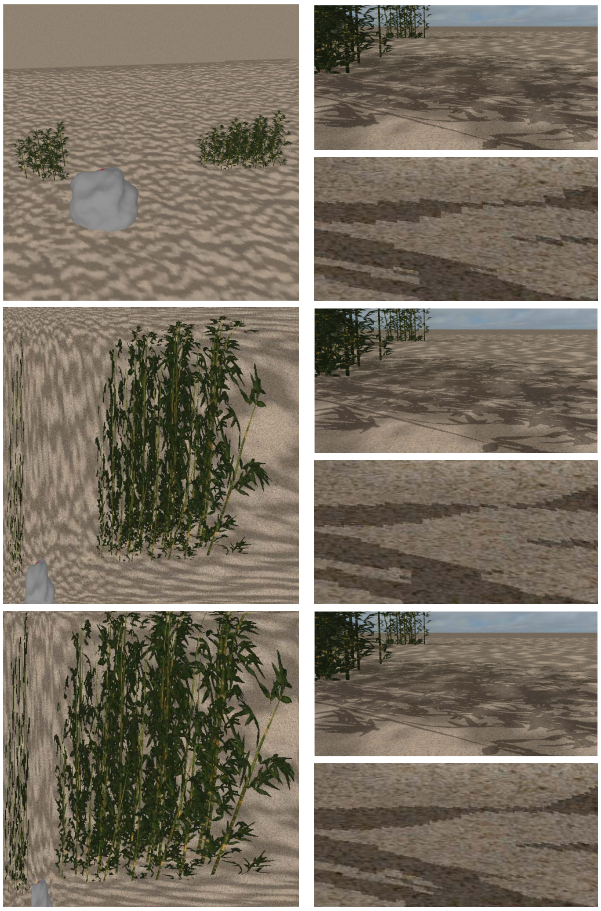


Figure 3: *The sandbox scene rendered using a 2,048×2,048 conventional shadow map (top, 2.5 ms), a 512×512 backwards RTW with desired view and distance importance functions (row 2, tessellation disabled, 3.3 ms, MER 2,114×4,616), and a 512×512 hybrid RTW using the desired view, distance, and edge functions (row 3, tessellation disabled, 3.5 ms, MER 2,783×5,628).*

Perspective Shadow Mapping (PSM) [Stamminger and Drettakis 2002; Wimmer et al. 2004; Chong and Gortler 2004; Lloyd et al. 2006; Lloyd et al. 2008] adaptively sample the scene in an effort to produce shadows with uniform aliasing artifacts. The shadow map applies a distortion to the light's view space to produce shadows which adapt to the user's viewpoint. RTW supports importance functions that produce results similar to PSMs. In contrast, PSMs only adapt to the user's viewpoint and do not account for other conditions, such as geometry features like surface normals, which might help improve shadow quality the way RTW does.

Adaptive Shadow Mapping (ASM) [Fernando et al. 2001; Lefohn et al. 2005] builds a tree which samples the scene at multiple resolutions that can adapt to the geometry, light, and viewing conditions. ASMs can produce high quality shadow results on a limited texture resolution budget. Building a complete ASM requires multiple passes on scene geometry in order to generate all of the nodes of the tree. Later advancements in ASMs have resulted in adaptive techniques which compute shadow maps using fewer iterative passes or no iteration at all [Giegl and Wimmer 2007a; Lefohn et al. 2007]. Shadow tests for ASMs may also be more expensive because they need to traverse the tree to find their associated node. A related technique, Queried Virtual Shadow Maps [Giegl and Wimmer 2007b], iteratively renders a dense shadow map as opposed to the sparse tree structure of the other ASM techniques. The biggest drawbacks to ASM approaches are that most need multiple rendering passes to produce a complete shadow map, and often the approaches lack sam-

pling coherency between cells of different scales. RTW avoids these limitations with single pass shadow map rendering and coherent changes in sampling rate.

Cascading Shadow Maps (CSMs) [Engel 2006; Lloyd et al. 2006] work by generating multiple shadow maps which cover increasingly larger portions of the scene. Generally, the desired view frustum is split into sub-regions each receiving its own shadow map. Similarly, Parallel-Split Shadow Maps [Zhang et al. 2006] subdivide the view frustum into sub-shadow maps. These techniques produce high quality shadows but suffer from a few limitations. A lack of sampling rate coherency can cause artifacts in the output when transitioning between different levels of the shadow map. Selecting the cutting planes can also be complicated—either done arbitrarily or requiring scene analysis for optimal placement. If cutting planes are not chosen well, errors can occur as the cutting planes move and shadow edges transition between neighboring cascades. Sample Distribution Shadow Maps [Lauritzen et al. 2011] are an example method which automatically places cutting planes by performing scene analysis on a single depth image rendered from the desired view. RTW addresses these limitations by providing sampling rate coherency removing the need for selecting cutting planes and replacing it with an importance map calculation. A further comparison between RTW and CSMs is performed in Section 6.4.

Finally, others have used hybrid approaches to improving shadow mapping quality. These approaches combine shadow mapping with volumetric shadows [Jensen and Keller 2004] or piecewise linear approximations of silhouettes [Sen et al. 2003] to improve the quality of shadows at the shadow edges. RTW could be used in conjunction with these methods to further improve its quality.

Sampling rate control and analysis is an important issue for shadow mapping. Almost all of the approaches mentioned above use some analytic approach to selecting sampling rate. In creating RTW, we believe that our mixture of analytic and heuristic importance functions provides developers more flexibility in controlling shadow resolutions. We are not the first to suggest such an approach. Tiled Shadow Maps [Arvo 2004], for example, used a heuristic-based analysis in the distribution of shadow map samples. For a detailed discussion about sampling rate analysis and more information on state-of-the-art real-time hard shadow mapping, we refer the reader to the article by [Scherzer et al. 2011] and the book *Real-Time Shadow* [Eisemann et al. 2011].

Our work uses image space warping to improve shadow quality. Warping has been previously used in applications such as terrain rendering [Dachsbacher and Stamminger 2004], occlusion reduction [Popescu and Aliaga 2006], and texture resampling [McGuire and Whitson 2008; Tarini and Cignoni 2005].

Contributions

RTW shadow maps are essentially a new approach to adaptive shadow mapping. The advantages of RTW shadow mapping are:

- RTW is a framework supporting any combination of analytic or heuristic functions for dynamically tuning shadow quality based upon changes in view, lighting, or scene geometry.
- RTW shadow maps are efficient to create. They require a small and fixed number of rendering and analysis steps, and they have a constant time shadow test.
- RTW produces shadow maps with sampling rate coherency producing shadows free of visible transitions in sampling rate.

3 Rectilinear Texture Warping

High throughput computer hardware excels at processing data stored in uniform grids with irregular data structures available only at a significant performance cost. Textures, being uniformly subdivided into texels, generally sample data in the same uniform pattern, although most data is non-uniform (i.e. regions of varying importance or features of different scales). This can cause a loss of information during encoding. Our approach decouples the uniform data structure from the sampling pattern while maintaining a high level of sampling coherency. This enables capturing data in an adaptive manner while maintaining high computational performance.

Warping Maps

To enable fast importance-based sampling, Rectilinear Texture Warped (RTW) images are composed of a conventional base texture enhanced with a rectilinear warping map, defined by 1-D warping maps for each major axis (vertical and horizontal). The warping maps are composed of a set of super-cells at equal or lower resolution than the base texture. The super-cells are uniformly sized and placed equidistant to one another in the input domain, and they point to monotonic locations in the output domain which maintains cell-to-cell coherency while still allowing adaptive sampling.

Figure 4 left shows an example 1-D warping map. Here, the base texture map is 32 texels wide and the warping map contains 8 super-cells. Each super-cell of the warp map contains a pointer to its output domain location, marked by the solid black and dashed orange lines. In this example, super-cell #1 is 4 texels wide in the input domain, but only 2 texels wide in the output domain, decreasing the sampling rate by 50% in this region. Super-cell #4 has the same 4 texel wide input domain with a 6 texel wide output domain, increasing the sampling rate by 50%.

Any data stored into a RTW texture is stored using its output domain coordinates. Texture coordinates in the output domain are easy to compute. Referring to Figure 4 left again, given the location of the purple **X** in the input domain coordinates, the output domain coordinate (dotted blue line) is found by performing a linear interpolation of the output domain coordinates between the 2 neighboring super-cells (marked by orange dashed lines).

Finally, 2 1-D warping maps, 1 vertical, 1 horizontal, are combined to form a rectilinear grid for storing 2-D texture data in non-uniform patterns. Figure 4 middle shows an example of 2 1-D warping maps combined to create a rectilinear grid. Storage for such a grid is inexpensive (the cost of storing 2 1-D warping maps) and is efficient to use (finding 2 warping values).

4 Rectilinear Texture Warped Shadow Maps

RTW shadow mapping uses an image space analysis process for identifying, adapting to, and sampling the regions of high importance. Initially, the scene is rendered and an importance map is calculated to identify sampling rate needs for various shadow map regions. The importance map is converted to a warping map which is used to render an RTW shadow map for the scene. Finally, the

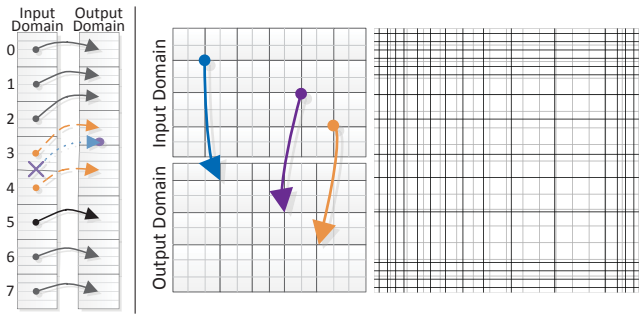
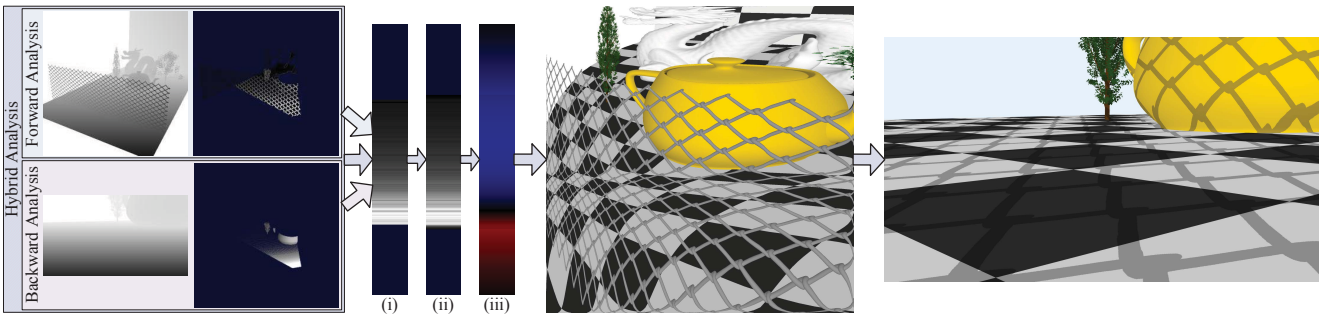


Figure 4: *Left: An example 1-D warping map used in RTW. Right: Two 1-D warping maps combined into a rectilinear grid (middle) and the rectilinear grid used for Figures 1 and 5 (right).*



(a) Importance Map (b) Warping Map (c) Shadow Map (d) Output Frame

Figure 5: *RTW shadow mapping pipeline visualized for the view rendered in Figure 1. In (a), (b.i), and (b.ii), the intensity of white indicates importance, while dark blue indicated unneeded data. In (b.iii), the color indicates direction of the warp, red for down, blue for up.*

output image is rendered or composited using the RTW shadow map as input. The algorithm, visualized in Figure 5, proceeds as follows.

Algorithm: RTW Shadow Mapping

(a) Build the importance map
(b) Convert 2-D importance map into 1-D warping maps
  (i) Collapse rows/columns to 1-D importance maps
  (ii) Blur importance maps
  (iii) Build warping maps from importance maps
(c) Render the RTW shadow map
(d) Render the output image from the desired view

4.1 Building the Importance Map

The warp maps are a simple yet powerful data structure, but they do not provide a convenient method of construction by themselves. To enable fast and easy construction, we specify 2-D importance maps which are transformed into warping maps. The 2-D importance map is specified as a regular grid along the light's image plane. Each grid cell receives a non-negative floating point importance $I_{uv} \in [0, \inf)$ where non-zero values imply a relative importance while zero implies unneeded data.

For creating importances maps, we present three possible approaches, forward, backward, and hybrid analysis. The forward and backwards methods analyze the scene from the light's view or desired view, respectively, while the hybrid approach combines these two analysis techniques. Each method has advantages and disadvantages, but we present all three in order to demonstrate the flexibility of RTW shadow mapping.

Forward Analysis The forward analysis method begins by rendering the scene from the light's perspective into a depth image whose resolution only needs to be large enough to detect the existence of the smallest features of interest. The importance map is then built by looking *forward* into the depth image for importance information. RTW shadow mapping supports any number of analytic or heuristic importance functions. The importance functions used in our implementation are enumerated in Section 5.

Backward Analysis The backward analysis method begins by rendering the output image with all shading information (except shadows), saving the color and depth. Importance is determined by projecting the output samples *backward* into the light's image space and analyzing their importance.

Hybrid Analysis As will be shown in Section 5, some importance functions work for both forward and backward analysis while some work only for one or the other. A final option is to combine both the forward and backward analysis methods. This has the advantage of giving the most flexibility but also comes with the highest associated computational cost.

4.2 Building Warp Maps

In the next step in the pipeline, the algorithm builds the RTW warping maps using the 2-D importance map as input to produce 1 row and 1 column warping map. (i) The importance for each cell is found by calculating the maximum importance within the respective row or column that each of the super-cells represent using $I_u = max(I_{u0}, I_{u1}, \dots)$ and $I_v = max(I_{0v}, I_{1v}, \dots)$. (ii) Once each super-cell has an importance value, the values are blended with neighbors using a Gaussian blur. This blur has 2 important effects. First, it produces a smooth transition in sampling rate, ensuring coherency. Second, the blur reduces the errors induced by non-linear rasterization (see Section 6.1). (iii) Finally, the warping map is built by shifting texel positions based upon the super-cell's weight relative to the total weight.

The displacement for a super-cell k of n total cells can be found using Equation 1, given the 1-D importance map I. Super-cell centers are moved based upon the ratio of neighbor importance to the total importance. Ultimately, the warping value is stored within the cell. During this process, super-cells which have been marked as unneeded (zero importance, dark blue) are implicitly culled away. This entire process is performed either on the CPU, or in our case, by performing the summations in pixel shaders. The rectilinear grid used for Figures 1 and 5 is shown in Figure 4 bottom right.

$$GetWarp(k) = \left(\sum_{j=1}^{k-1} I_j / \sum_{j=1}^{n} I_j \right) - \left(\frac{k}{n} \right) \qquad (1)$$

4.3 Rendering the Shadow Map

Step (c) renders the RTW shadow image. Rendering proceeds by computing an output domain position for every vertex. First, the vertex V is projected with the view and projection matrices (M_L) used for the conventional shadow map to the location P using $P = M_L \cdot V$. The projected location P is used to locate the vertical and horizontal super-cells in the warp maps, and the vertex is relocated to its output image plane location P' by applying the warp, $P'_x = P_x + GetWarpX(P_x)$ and $P'_y = P_y + GetWarpY(P_y)$[1]. Finally, the triangles are rasterized (see Section 6.1).

4.4 Producing the Output Frame

The output image is finally produced in step (d). The assembly of the final output frame comes in two forms. In the case of the forward analysis method, the scene geometry must be drawn one additional time and shadowed using the RTW shadow map. For the backward and hybrid analysis methods, the output image frame has already

[1]It should be noted that P_x and P_y are in clip space $[-1, 1]$ while the $GetWarp$ function assumes texture space $[0, 1]$. As such, $GetWarpX$ and $GetWarpY$ need to be adapted to clip space.

been computed in step (a). Therefore, only the shadow needs to be calculated and composited on top of the image.

The calculation of shadows proceeds in much the same way as conventional shadow mapping. The conventional shadow map texture coordinates (s,t) are first calculated. RTW shadow map coordinates are found by warping the coordinates (s,t) to (s',t') where $s' = s + GetWarpS(s)$ and $t' = t + GetWarpT(t)$.

5 Importance Functions

RTW support any set of analytic or heuristic functions by way of the importance map. We discuss 4 functions in this work: samples in desired view, distance to eye, shadow edges, and surface normal. Given a texel (u,v), the output importance is calculated for m importance functions by taking the product of the function values $I(u,v) = \prod_{i=1}^{m} I_i(u,v)$. This permits an interdependence between importance functions giving any importance function the ability to consider a texel unimportant (i.e. excluded from the shadow map, the dark blue pixels in Figure 5), of reduced importance, or of increased importance.

5.1 Desired View Function

Shadows only need to be calculated for samples falling within the desired view (DV). For this reason, the importance function I_{DV} returns 1 for any point within the desired view and 0 for all other points. This method provides focusing for the shadow map but does nothing else to improve shadow quality.

This function could be used on both the forward and backward analysis methods, though using this method with forward analysis can lead to errors when a large shadow casting object occludes regions of the desired view from the light, potentially causing these regions to be unsampled. This error does not exist when performing backward analysis and is therefore the recommended usage.

5.2 Distance to Eye Function

With a perspective projection for the desired view, shadow map regions farther from the eye of the desired view will have lower sampling needs since they are smaller on the output image plane. The distance to eye function, accommodates this feature. Given a sphere that is q distance from the eye along the view direction, the size of the projected sphere will be proportional to $1/q$.

Since the depth value produce by the projection matrix is analogous to $1/q$, that is exactly the value we use. For forward analysis, the importance value is found by transforming the texel from the light space (M_L) to the desired view space (M_D), $I_D(u,v) = 1 - \left(M_D \cdot M_L^{-1} \cdot [u,v,depth(u,v)]'\right)_z$. For backward analysis, the importance can found using the depth directly with $I_D(u,v) = 1 - depth(u,v)$. The resulting function will have importance values in the range $[0,2]$ for valid points. Points outside of the view frustum can be removed by combining this function with the desired view function.

Figure 3, row 2, shows an example texture map that uses this function. The quality of the shadow at 512×512 is still better than that of conventional shadow mapping at 2,048×2,048 resolution. Still, some texture regions are underutilized.

5.3 Shadow Edge Function

Shadow edges, the transition region between in-light and in-shadow, are the most notable feature of shadows with inadequate sampling. The next importance function, $I_E(u,v)$, increases the sampling rate for these regions. This function provides additional sampling in regions which contain shadow edges by reducing the sampling in regions without edges. This method *only* works using forward analysis and is done by performing a depth discontinuity test on each texel of the depth image with its eight neighboring texels. This is done by taking the difference between 2 texels and comparing the value to the shadow bias. Texels with shadow edges have their full importance included while other regions receive a minimal importance α (in our examples $\alpha = 0.001$).

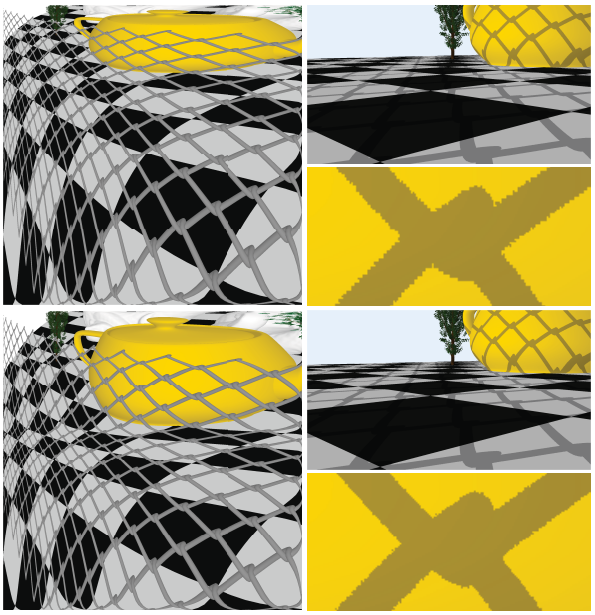


Figure 6: *RTW example using backward analysis with the desired view and distance functions (top, tessellation disabled, 5.4 ms, MER 11,451×20,761). The surface normal function is added (bottom, tessellation disabled, 4.2 ms, MER 11,447×16,333) increasing the sampling of the teapot in the shadow map (left) resulting in a marginally better shadow (right).*

Figure 3, row 3, shows the edge function combined with the distance function. The additional function reduces the sampling in areas without any edges and captures finer details than the distance function alone.

5.4 Surface Normal Function

A surface at some fixed distance away from the desired view has a maximal projected screen space size when pointing towards the view direction (vd). As the surface is rotated around its centroid, the projected screen space surface area becomes smaller until it reaches zero when the surface is perpendicular to the rays sampling it. The final function used is an approximation to this quality of surfaces. Simply enough, it is based upon the dot product between the surface normal and the view direction, $I_{SN}(u,v) = 1 + \beta \cdot clamp(-normal(u,v) \cdot vd, 0, 1)$.

Here, the function gives bonus importance β (in our implementation $\beta = 2.0$) to surfaces pointing towards the view direction while not specifically penalizing those pointing away. This function works for all of the proposed analysis techniques. Figure 6 top shows the shadow calculated with only the distance and shadow edge functions. When the surface normal function is added in Figure 6 bottom, the sampling of the teapot surface increases as does the quality of the shadow in that region.

5.5 Maintaining Temporal Coherency

RTW only provides sampling coherency and the importance functions defined thus far do not explicitly provide any temporal co-

Table 1: *Range of frame rendering times for a typical path through the 3 scenes used to demonstrate the RTW shadow mapping system.*

	Playground	Sandbox	Town
Conventional	2.7-3.6 ms	1.9-2.1 ms	2.7-3.2 ms
Forward Analysis (Dynamic Tessellation)	5.3-7.0 ms (7.2 - 8.4 ms)	4.2-5.3 ms (5.8-7.2 ms)	3.7-4.8 ms (11.0-11.8 ms)
Backward Analysis (Dynamic Tessellation)	4.2-5.9 ms (6.2-7.3 ms)	4.0-5.4 ms (6.2-7.1 ms)	3.7-4.8 ms (11.0-11.5 ms)
Hybrid Analysis (Dynamic Tessellation)	5.8-7.3 ms (7.8-8.9 ms)	5.0-5.7 ms (6.5-7.2 ms)	4.5-5.4 ms (11.8-12.5 ms)

herency in frame-to-frame shadow results. If the changes in the geometry, view, and lighting conditions are temporally coherent, the output shadow result should be temporally coherent as well. To provide a guarantee of temporal coherency, the current and previous importance maps can be blended together. In our examples, we found this temporal blending unnecessary, so we mention it only for completeness.

6 Results and Discussion

All experiments were run on a PC with an Intel 3.2 GHz i7 processor, 12 GB of RAM, and an nVidia GeForce GTX 480. RTW is implemented using nVidia Cg shader version 5.0. All results are reported for 1080p output resolution (1920×1080). The resolution of the depth image used for step (a) of forward and hybrid analysis techniques was always 512×512. All importance maps were 512×512, leading to warping maps that were 512 super-cells wide. All examples use shadow maps of 2,048×2,048 unless otherwise noted. Most figures are marked with a *maximum effective resolution (MER)* for the RTW shadow map. This is the conventional shadow map resolution required to match the resolution of the largest RTW super-cell. The shadow bias value was fixed by hand, though their is certainly the opportunity to adapt the bias based upon the scene analysis results.

Our system was tested on 3 scenes: playground (Figures 1, 5, 6, 7, and 8), sandbox (Figure 3), and town (Figure 10). The scenes consisted of 712k triangles for playground, 28k triangles for sandbox, and 528K triangles for the town scene. Rendering performance in the various scenes is outlined in Table 1. Geometry rendering was not optimized, besides the use of vertex and index buffers. All stages of the RTW pipeline were run on the GPU in vertex, pixel, or tessellation shaders. Table 2 shows the time spent in each of the various stages of the processing pipeline.

6.1 Rendering RTW Images

The image plane distortion introduced by the warping maps produces triangles that may potentially have curved edges, requiring non-linear rasterization for accurate shadows. A general method for non-linear rasterization is an open problem [Gascuel et al. 2008; Popescu et al. 2010], where a number of possible solutions including point-based rendering, adaptive subdivision, and raycasting are used. The recent availability of tessellation hardware on graphics platforms makes subdivision the most reasonable choice in our eyes.

Table 2: *List of processing time needed to execute each stage of the RTW pipeline for the view in Figure 1.*

	Conventional	Forward Analysis	Backward Analysis	Hybrid Analysis
(a) Importance Map Calculation	---	1.3 ms	1.4 ms	2.7 ms
(b) Warping Map Calculation	---	0.1 ms	0.3 ms	0.4 ms
(c) Shadow Map Rendering (w/ tessellation)	1.3 ms	2.2 ms (3.6 ms)	2.2 ms (3.9 ms)	2.0 ms (3.6 ms)
(d) Output rendering or compositing	1.8 ms	1.9 ms	0.9 ms	1.1 ms
Total	3.1 ms	5.5 ms (6.9 ms)	4.8 ms (6.5 ms)	6.2 ms (7.8 ms)

Table 3: *Rendering time for backward analysis RTW using a tessellation shader for triangle subdivision. The percentages represent the maximum relative length of a triangle edges to the shadow map width (i.e. no edge is more than 5% of the shadow map width).*

	Disabled	5%	2.5%	1%
Playground	4.3-5.4 ms	5.5-6.9 ms	5.8-7.0 ms	6.0-7.2 ms
Sandbox	3.8-4.8 ms	5.2-5.9 ms	5.9-7.0 ms	8.0-9.5 ms
Town	3.8-5.0 ms	8.4-9.5 ms	10.9-11.9 ms	16.5-17.4 ms

An accurate subdivision algorithm would take each triangle, project it to the screen, and subdivide for each vertical and horizontal super-cell it crosses, a worst-case $\mathbb{O}(tn^2)$ operation for t triangles and n super-cells per direction. By using a divide and conquer approach, the process can be reduced to an $\mathbb{O}(t\ log\ n)$ operation. For interactive applications with large numbers of triangles, this is still too expensive of an operation. An approximate solution is to just tessellate triangles based upon their screen space size.

Our implementing uses on-the-fly subdivision via the tessellation shader capabilities now available on GPUs. The tessellation proceeds as follows. Project the triangle with RTW. Set tessellation factors based upon edge length, such that no edge is above ϵ in length. Tessellate triangles and reproject with RTW. The flexibility afforded by tessellation shaders requires this operation only happen once, outputting triangles that all have screen space edges less than ϵ. Table 3 shows the tessellation performance for various maximum edge length settings. In practice, the 2.5% setting struck a good balance between performance and image quality. Figure 7 shows a typical example of the errors that can occur without tessellation.

The quantity of tessellation needed is also impacted by the smoothness of sampling rate variation. Smoother changes will cause fewer errors in under tessellated data, while sharper variation will produce larger errors. Therefore, the Gaussian blur computed in step (b.ii) impacts the quantity of tessellation needed. Increasing the σ of the Gaussian gives smoother transitions at the cost of reduced sampling flexibility and wasted shadow samples. For all of our examples, $\sigma = 3$ texels.

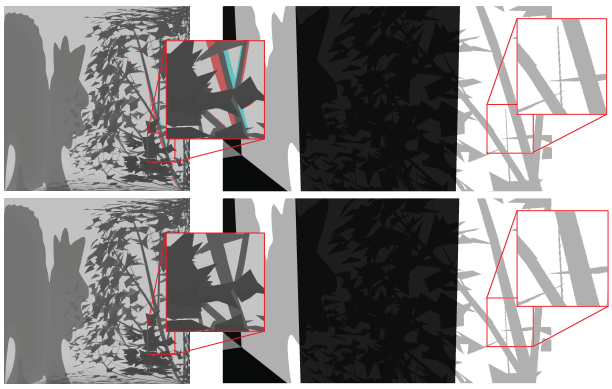


Figure 7: *Example shadow (right) using tessellation for RTW (MER 19,448×36,959) where the shadow maps (left) are differenced with the ground truth. No tessellation (top, 5.0 ms) leads errors to appear in the shadow map. With triangles tessellated to be no larger than 2.5% of the shadow map (bottom, 6.5 ms), most errors are corrected.*

Figure 8: *A backward RTW shadow map at 1,024×1,024 is uses a single texel shadow test (left, tessellation disabled, 4.8 ms, MER 11,614×13,650) or a 4 texel wide jittered PCF to either antialias the shadow (middle, 7.3 ms) or to create a soft shadow (right, 7.3 ms).*

6.2 Percentage Closer Filtering

Percentage closer filtering (PCF) [Reeves et al. 1987] is an approach often used to enhance the quality of shadows by antialiasing the shadow edges or for simulating soft shadowing effects. PCF samples the shadow map at multiple locations using a Gaussian weighting across a set of jittered samples or a regular grid of samples. The RTW shadow map can be easily integrated with PCF. To produce antialiased shadow edges, shadow map coordinates are first warped. Those warped coordinates are fed to the PCF where the shadow value is calculated. Using RTW shadow maps improves shadow edge results by implicitly calculating shadow edges at higher frequencies because of the importance-based sampling of the RTW shadow map. In Figure 8 middle, a 2-pixel jittered Gaussian PCF is used to render the shadows making for cleaner looking edges. To simulate soft shadow effects, the area of the PCF in light space is important to maintain. To keep the area constant, unwarped shadow map coordinates are fed to the PCF function. The PCF function is modified to then warp the shadow map coordinates at the last possible moment. Figure 8 right demonstrates this approach using an RTW shadow map. Using PCF tended to reduce performance by 0.25 ms to 1.0 ms for each increase in radius of influence for *both* conventional and RTW shadow mapping.

6.3 Limitations

There are a few basic limitations with both the forward and backward analysis RTW approaches. With forward analysis, if the initial depth image resolution is too low, important details could potentially be missed and excluded from the importance map. This problem can be addressed by rendering the depth image at higher resolution, but knowing what size to use can be a difficult problem. With the backward analysis method, different desired view points may project to the same importance map location, with different importance values. Currently this problem is addressed by only keeping the highest value point, but other situations might demand different approaches.

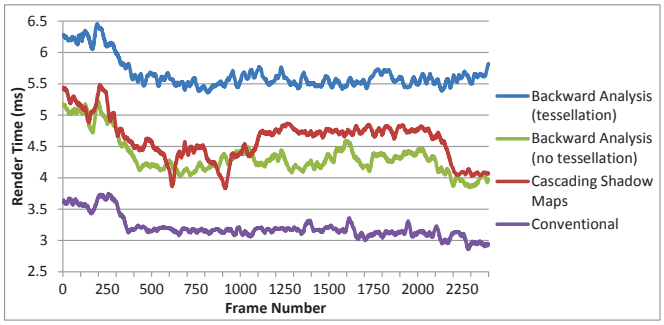


Figure 9: *Timing comparison for a typical path through the town scene. Results are takes over approximately 2400 frames and recorded in milliseconds.*

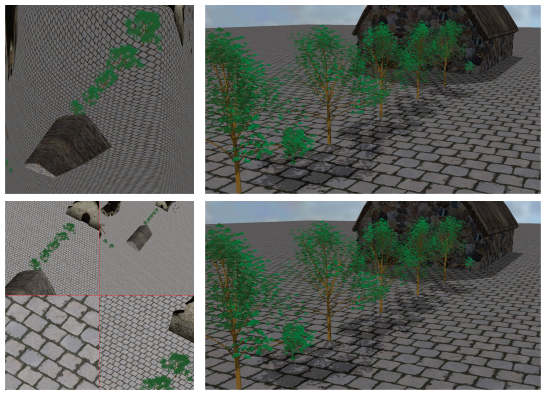


Figure 10: *Shadow maps (left) and output images (right) shown for a view from the performance statistics in Figure 9 using backward analysis RTW (top, tessellation enabled) and CSM (bottom).*

The use of a rectilinear grid for distortion has limitations of its own. If one single element requires additional resolution, it forces all elements in the same row and column to receive extra resolution, even if unneeded. In extreme situations this can lead to reduction in overall quality. Even so, in our experiments we never experienced a configuration which led to visible quality issues. Initially, we had explored a similar approach replacing the rectilinear grid with a 2-D grid connected by a spring-mass system. While the spring-mass system adapted locally to provide additional resolution, it lacked the ability to globally reallocate resolution in the same way that our rectilinear warping system does. Comparing the two approaches, we quickly came to the conclusion that the rectilinear warping was far superior to spring-mass system in both quality and computational performance.

The non-linear rasterization remains the most difficult limitation of RTW to overcome. If triangles are insufficiently tessellated, they may create visual artifacts in the output along the shadow edges. That said, the number of triangles used in modern graphics applications is rising while the size of those triangles is shrinking. Given that the level of tessellation required for RTW shadow maps is relatively limited, we feel this remains only a short term issue.

6.4 Comparison to CSM

Cascading Shadow Maps (CSM) [Engel 2006; Lloyd et al. 2006] produces high quality shadow maps by rendering smaller shadow maps of multiple regions based upon the user's position and view direction. The advantage of this technique is that it uses a fixed number of conventional rasterization steps to generate shadow maps and produces shadows of comparable quality to RTW. Figure 9 compares the timing information for a motion path through the town scene. The comparison includes conventional shadow mapping, backward analysis RTW (both with and without dynamic tessellation of 5%), and a basic CSM implementation which uses 4 cascades (Figure 10). In this case, CSM outperforms RTW with tessellation but performs similarly when tessellation is disabled. This is of course very algorithm, scene, and configuration dependent, but in this case, RTW performs on par with CSM in both performance and quality. The advantages of RTW over CSM remain greater sampling flexibility and by accommodating the perspective aliasing which CSM ignores, eliminates the risk of sampling rate discontinuities when crossing cascades.

7 Conclusion

We have presented RTW, a system for producing single image adaptive shadow maps. The RTW system is efficient, requiring some overhead beyond conventional shadow mapping, yet it produces

shadows whose quality approaches that of raytracing. RTW features simple and fast construction making it perfect for interactive exploration of dynamic 3-D scenes. This is achieved by using rectilinear warping to decouple the sampling rate from the underlying uniform data structure. At the same time, RTW shadow maps have sampling coherency making variation in the sampling rate seamless to any configuration of user position, light position, or scene geometry.

One issue considered but never pursued was the impact of rotating the light's image plane, and thus modifying the 2 1-D importance maps. The orientation of the importance maps certainly would affect the utilization of samples. An optimization of this rotational component would likely result in even better shadow quality.

We have presented four importance functions for determining sampling rate within the context of two analysis pipelines. The backward analysis pipeline showed better computational performance while the hybrid pipeline allows more flexibility in determining importance. These are not the only four functions which can be used. RTW allows for a wide variety of additional functions. For example, shadows near the center of the view frustum or those cast by an important character might receive a greater number of samples. Even analysis of the output pixel luminance after shading could impact the importance of the shadow pixel (i.e. pixels which are almost black anyways need less detailed shadow information).

Acknowledgments

This work was performed under National Science Foundation award CDI-0835821 and NIH/NCRR Center for Integrative Biomedical Computing, 2P41 RR0112553-12. We would also like to thank the Voicu Popescu and Josh Levine for useful feedback, in addition to all of the reviewers.

References

AILA, T., AND LAINE, S. 2004. Alias-free shadow maps. In *Rendering Techniques 2004*, 161–166.

ARVO, J. 2004. Tiled shadow maps. In *Proceedings of the Computer Graphics International*, 240–247.

BRABEC, S., ANNEN, T., AND PETER SEIDEL, H. 2000. Practical shadow mapping. *Journal of Graphics Tools 7*, 9–18.

CHONG, H., AND GORTLER, S. J. 2004. A lixel for every pixel. In *Eurographics Symposium on Rendering*, 167–172.

CROW, F. C. 1977. Shadow algorithms for computer graphics. *SIGGRAPH Comput. Graph. 11*, 2, 242–248.

DACHSBACHER, C., AND STAMMINGER, M. 2004. *Rendering Procedural Terrain by Geometry Image Warping*. 103110.

EISEMANN, E., SCHWARTZ, M., ASSARSSON, U., AND WIMMER, M. 2011. *Real-Time Shadows*. Taylor and Francis.

ENGEL, W. 2006. Cascaded shadow maps. In *ShaderX5*. Charles River Media.

FERNANDO, R., FERNANDEZ, S., BALA, K., AND GREENBERG, D. P. 2001. Adaptive shadow maps. In *SIGGRAPH '01*, 387–390.

GASCUEL, J.-D., HOLZSCHUCH, N., FOURNIER, G., AND PÉROCHE, B. 2008. Fast non-linear projections using graphics hardware. In *I3D '08*, 107–114.

GIEGL, M., AND WIMMER, M. 2007. Fitted virtual shadow maps. In *Graphics Interface '07*, 159–168.

GIEGL, M., AND WIMMER, M. 2007. Queried virtual shadow maps. In *I3D '07*, 65–72.

JENSEN, H. W., AND KELLER, A., 2004. Abstract an efficient hybrid shadow rendering algorithm.

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*, 1462–1482.

LAURITZEN, A., SALVI, M., AND LEFOHN, A. 2011. Sample distribution shadow maps. In *I3D '11*, 97–102.

LEFOHN, A., SENGUPTA, S., KNISS, J., STRZODKA, R., AND OWENS, J. D. 2005. Dynamic adaptive shadow maps on graphics hardware. In *ACM SIGGRAPH '05 Sketches*.

LEFOHN, A. E., SENGUPTA, S., AND OWENS, J. D. 2007. Resolution-matched shadow maps. *ACM Trans. Graph. 26*.

LLOYD, D. B., TUFT, D., YOON, S.-E., AND MANOCHA, D. 2006. Warping and partitioning for low error shadow maps. In *Eurographics Symposium on Rendering*, 215–226.

LLOYD, D. B., GOVINDARAJU, N. K., QUAMMEN, C., MOLNAR, S. E., AND MANOCHA, D. 2008. Logarithmic perspective shadow maps. *ACM Trans. Graph. 27*, 106:1–106:32.

MCGUIRE, M., AND WHITSON, K. 2008. Indirection mapping for quasi-conformal relief mapping. In *I3D '08*.

PAGOT, C., COMBA, J., AND DE OLIVEIRA NETO, M. 2004. Multiple-depth shadow maps. In *SIBGRAPI 2004*, 308 – 315.

POPESCU, V., AND ALIAGA, D. 2006. The depth discontinuity occlusion camera. In *I3D '06*, 139–143.

POPESCU, V., ROSEN, P., ARNS, L., TRICOCHE, X., WYMAN, C., AND HOFFMANN, C. M. 2010. The general pinhole camera: Effective and efficient nonuniform sampling for visualization. *IEEE TVCG 16*, 777–790.

REEVES, W. T., SALESIN, D. H., AND COOK, R. L. 1987. Rendering antialiased shadows with depth maps. In *SIGGRAPH '87*, 283–291.

SCHERZER, D., WIMMER, M., AND PURGATHOFER, W. 2011. A survey of real-time hard shadow mapping methods. *Computer Graphics Forum 30*, 1, 169–186.

SEN, P., CAMMARANO, M., AND HANRAHAN, P. 2003. Shadow silhouette maps. In *SIGGRAPH 2003*, 521–526.

SINTORN, E., EISEMANN, E., AND ASSARSSON, U. 2008. Sample based visibility for soft shadows using alias-free shadow maps. *Computer Graphics Forum 27*, 4, 1285–1292.

STAMMINGER, M., AND DRETTAKIS, G. 2002. Perspective shadow maps. In *SIGGRAPH '02*, 557–562.

TARINI, M., AND CIGNONI, P. 2005. Pinchmaps: Textures with customizable discontinuities. eurographics. *Computer Graphics Forum*, 557–568.

WHITTED, T. 1980. An improved illumination model for shaded display. *Commun. ACM 23*, 343–349.

WILLIAMS, L. 1978. Casting curved shadows on curved surfaces. In *SIGGRAPH '78*, 270–274.

WIMMER, M., SCHERZER, D., AND PURGATHOFER, W. 2004. Light space perspective shadow maps. In *Eurographics Symposium on Rendering*.

WOO, A. 1992. *The shadow depth map revisited*. Academic Press Professional, Inc., San Diego, CA, USA, 338–342.

ZHANG, F., SUN, H., XU, L., AND LUN, L. K. 2006. Parallel-split shadow maps for large-scale virtual environments. In *ACM Virtual Reality Continuum and Its Applications*, 311–318.