High-Performance Graphics 2023
J. Bikker and C. Gribble
(Editors)

COMPUTER GRAPHICS *forum*
Volume 42 (2023), Number 8

Surface Light Cones - Supplementary Material

Pascal Stadlbauer[1], Alexander Weinrauch[1], Wolfgang Tatzgern[1], Markus Steinberger[1,2]

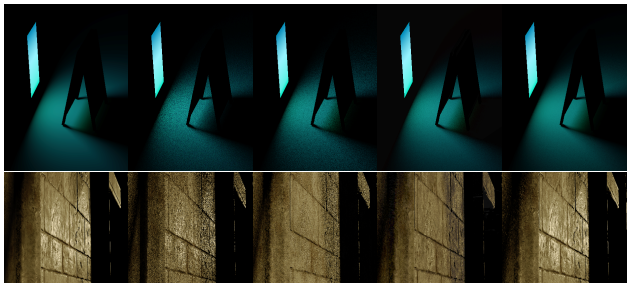
[1]Graz University of Technology, Institute of Computer Graphics and Vision, Austria
[2] Huawei Technologies

1. Alternative cache entries

Of course, there are other alternative representations, than our cone representation, that could be used to approximate L_i in a cache. Arguably, one of the most straight-forward approaches would be a constant approximation of L_i for each light source, by e.g. extending the hard shadow binary cache [WTS*23]: For each area light source, we could store one floating point number that represents its partial visibility which we could progressively update using ray tracing. For shading, we could approximate L_i using the stored visibility and concentrate all radiance to the center direction of the light source. Such a simple approach not only lacks detailed directional information during shading, but also suffers for non-uniform light sources, as shown in Figure 1 top. Note how our cone representation not only better captures the light intensity, but also captures which parts of the area light source are visible from which surface points, i.e., that the area under the chair receives more green light than blue.

Another commonly used representation for capturing radiance are spherical harmonics (SH) [Mül06]. SH can not only be used to capture L_i, but could also be used for L_o. In theory, an SH basis could faithfully capture either and could thus serve as a perfect cache. However, to represent high frequencies, a large number of bands would be necessary, which would not only lead to very high memory requirements, but also computing and updating all those bands would require a high amount of compute. Thus, we present experiments with using three SH bands only, which results in 27 values per cache entry (compared to our 18). As shown in Figure 1 bottom, SH for L_o is not capable of capturing view-dependent effects well. Using SH for L_i can capture some view dependent effects, but these heavily depend on the sampling and thus introduce significant amounts of noise.

For a quantitative evaluation of the different representations, see Table 1. To reach a somewhat fair comparison, we tested on our completely static version of Sponza only and let all approaches run until convergence. In this way, we evaluate the ceiling of the approaches, ignoring that cones converge with the lowest number of samples and require significantly fewer computations compared to SH. SHs cannot capture high frequencies of L_o; caching L_i (as all other 3 methods) clearly increases quality.



(a) *GT* **(b)** *Vis* **(c)** *SH* L_i **(d)** *SH* L_o **(e)** *Ours*

Figure 1: *Using a visibility term per light source (b) is a meager representation for close-by non-uniform light sources (top) and leads to noisy evaluations for view dependent shading as area light sources are contracted to a single point (bottom). Spherical harmonics as alternative cache representation for L_i (c) and L_o (d) (each using 27 coefficients) are able to represent smooth shading (top) but cannot capture high frequency details as needed for highly specular surfaces (bottom). Our cones (e) only require 18 values, but are significantly better suited to capture the incoming radiance from area light sources, create view-dependent detail (bottom) and accurate capture the color gradients for non-uniform light sources (top).*

	Flip↓	PSNR↑	SSIM↑
SH L_o	0.329	18.45	0.847
SH L_i	0.151	28.35	0.966
Visibility	0.119	27.84	0.958
Cones	0.058	33.19	0.978

Table 1: *Quality comparison between alternative cache entry representations for Sponza, while running to convergence. Although our approach converges with the fewest number of samples, it also achieves the best quality metrics with a significant margin.*

2. Cache Bias Comparison

Adding a bias to the cache allocation essentially reduces the cache resolution and forces spatial reuse of the same cache data. While a higher bias may significantly increase performance, it also may

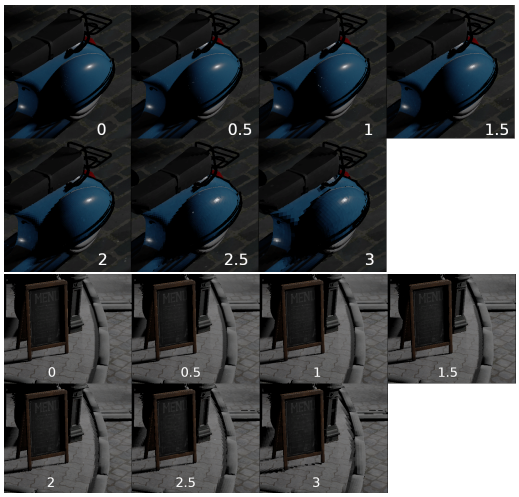


Figure 2: *Bias comparison details for Bistro Exterior [Lum17]: for a cache bias of up to* 1.5 *visual quality is close to identical to for resolution caching. With a bias of* 2.0 *and more artifacts around object boundaries and shadow boundaries become visible.*

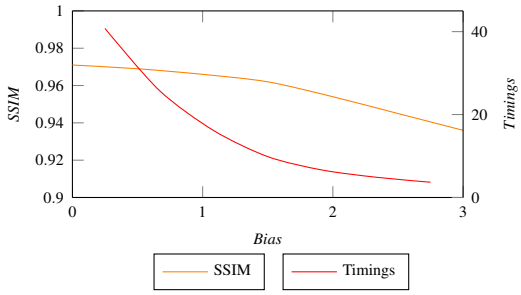


Figure 3: *Bias comparison for Bistro Exterior: Increasing the bias of our caches significantly reduces rending times (in ms), however, quality stays hardly reduces up to a bias value of* 1.5. *Thus, our bias of* 1.0 *is still conservative.*

reduce quality similar to a shadow map resolution that is too small. We plot the relation between quality and rendering time in Figure 3, again for a static converged scene. Clearly, a bias of up to 1.5 hardly changes quality. As such, our test setup using a bias of 1.0 is still very conservative and we could have reduced rendering times by another 30% with little impact on quality. Example renderings for the different bias values are shown in Figure 2. Even for this difficult scene parts, a bias of 1.5 is hardly distinguishable from 0.0. A bias of 2.0 leads to visible artifacts around the edges of objects and a higher bias clearly shows artifacts for hard shadows and at boundaries.

3. Speed and Size of lights

Edge cases are evaluated in two setups. The first setup 4 includes a emissive sphere moving over 100 units in 10 seconds. Next to its path there are obstacles that cast shadows. We speed up the simulation in order to look how results differ with changing speeds.

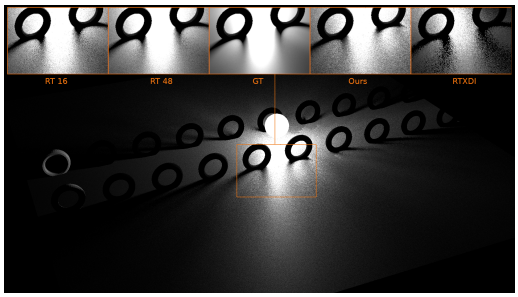


Figure 4: *We show an edge test case of a fast light, that moves over 100 units in 0.625s (16x speedup). Our method develops more noise than usual and show a slight lag in radiance.*

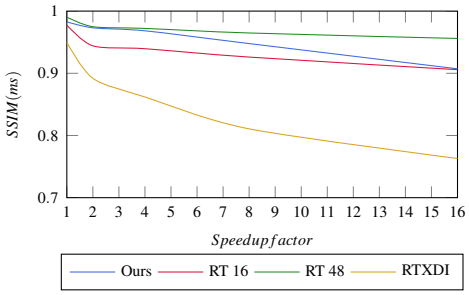


Figure 5: *Comparison for a light moving at different speeds over 100 units. For 1x speed the light takes 10s to move 100 units*

One can see that for very high speeds noise starts occurring and a slight lag shows. This lag can of course be combated by using more samples during cone adaption.

To get a feeling how the size of light sources impacts results we scale a light from one square units to a hundred square units. This can be seen in figure 6. For larger sizes noise starts to occur, but we still achieve good results.

4. Additional Multiviewer Timings

To further show effectiveness we show two setups with 60 percent overlap in figure 8. One for our Bistro Exterior (based on [Lum17]) setup and one for our Sponza setup (based on [Int]).

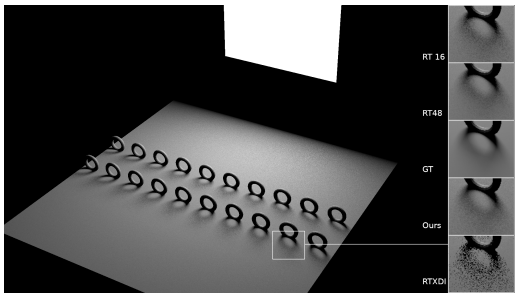


Figure 6: *We show a test case of a light scaling from* $1m^2$ *to* $100m^2$ *in 10 seconds. Here seen at about 50 m2.*

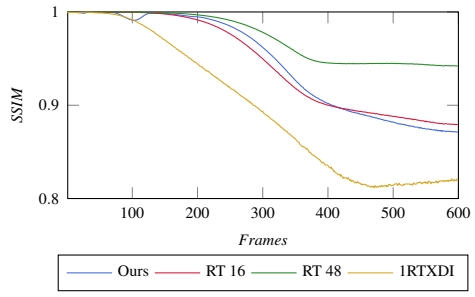
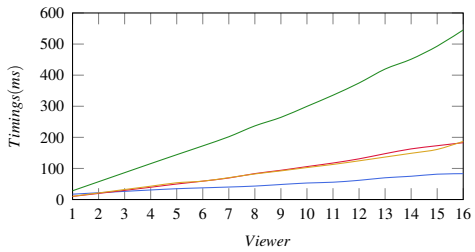
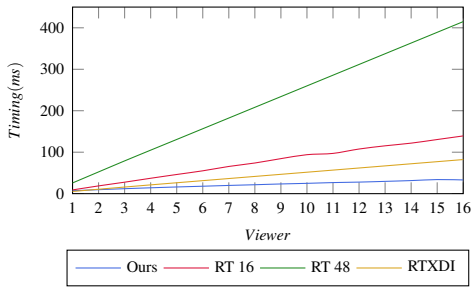


Figure 7: *Light Scaling from* $1m^2$ *to* $100m^2$ *as seen in 6.*



(a) *Bistro Exterior*



(b) *Sponza*

Figure 8: *Scaling for 1 to 16 viewers in Bistro Exterior and Sponza for 60 percent overlap demonstrating our superior scaling in comparison to screen-space approaches. Note that our approach also achieves superior quality.*

References

[Int] INTEL: Intel's sponza. https://www.intel.com/content/www/us/en/developer/topic-technology/graphics-research/samples.html. Accessed: 2023-03-16. 2

[Lum17] LUMBERYARD A.: Amazon lumberyard bistro, open research content archive (orca), July 2017. URL: http://developer.nvidia.com/orca/amazon-lumberyard-bistro. 2

[Mül06] MÜLLER C.: *Spherical harmonics*, vol. 17. Springer, 2006. 1

[WTS*23] WEINRAUCH A., TATZGERN W., STADLBAUER P., CRICKX A., HLADKY J., COOMANS A., WINTER M., MUELLER J. H., STEINBERGER M.: Effect-based multi-viewer caching for cloud-native rendering. *ACM Trans. Graph. 42*, 4 (jan 2023). 1