Implicit Sphere Shadow Maps

Michael Krone* Guido Reina† Sebastian Zahn Tina Tremel Carsten Bahnmüller Thomas Ertl‡

Visualization Research Center, University of Stuttgart, Germany

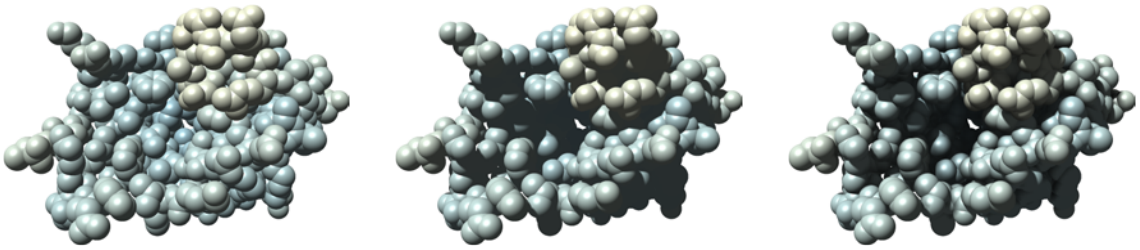

Figure 1: Local lighting often cannot sufficiently highlight the shape of particle data sets (left), which can encumber the visual analysis. We present a method to create soft shadows for particle data sets using Implicit Sphere Shadow Maps, which enhances the perception of the shape of particle data sets (center). Directional shadows mostly convey larger-scale structural information. Our shadow map method can be combined with real-time Ambient Occlusion (AO), which is commonly used in scientific visualization (right). We use AO to recover spatial information in completely shadowed regions and add it in completely unshadowed ones.

Abstract

Particle data are commonly visualized by rendering a sphere for each particle. Since interactive rendering usually relies on fast local lighting, the spatial arrangement of the spheres is often very hard to perceive. That is, larger functional structures formed by the particles are not easily recognizable. Using global effects such as ambient occlusion or shadows adds important depth cues. In this work, we present Implicit Sphere Shadow Maps (ISSM), an application-tailored approach for large, dynamic particle data sets. This approach can be combined with state-of-the-art object-space ambient occlusion to further emphasize the spatial structure of molecules. We compare our technique against state-of-the-art methods for interactive rendering with respect to image quality and performance.

Index Terms: Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture Computer Graphics [I.3.8]: Applications—Molecular Visualization

1 Introduction

For the interactive visual exploration and analysis of large particle data (e.g., molecular dynamics or SPH simulation data), a faithful visualization is necessary that makes structures and features contained in the data visible. A glyph-based rendering that represents each particle by a sphere is commonly used and can be seen as the 'ground truth' rendering of the data. For analysis, local lighting alone is often not sufficient because the spatial arrangement of the spheres in relation to each another is difficult to see or not even discernible at all. Furthermore, depth cues are hardly present. By introducing shadows, data set features become visible and can be analyzed.

Tarini et al. [15] presented a combination of several interactive rendering techniques to enhance depth and shape perception. They used ambient occlusion, self-shadowing, fogging, and highlighted edges to emphasize the structure of molecular data sets. Their work was influential for subsequent developments in molecular visualization. However, the methods they employed partially require preprocessing and are, thus, only interactive for static data.

We employ a combination of ambient occlusion and real-time shadows for an interactive visualization of higher-level structures in particle data sets. This enhances the spatial perception and allows users to analyze features of particle data sets especially for the purpose of scientific research. Thus, we propose to use implicit shadow maps to improve the spatial perception of large-scale features in the data. Our approach is tailored for single-GPU COTS desktop workstations. Our method is conceptually similar to hybrid ray/frustum traced shadows [13, 14]. However, since our scene consists of implicitly represented surfaces, we can use this information to obtain a higher precision at lower memory cost compared to conventional, tessellated objects. Similar to previous work, our technique can also render soft shadows. An example is shown in Fig. 1.

2 Related Work

Physically-based shadows are an important cue for depth and spatial arrangement of objects in a scene. Traditional ray tracing enables rendering of hard-edged shadows by casting a *shadow ray* from a given point in the scene towards every light source. If any shadow ray intersects any geometry in the scene, the point lies in shadow with respect to this light [6, 16]. However, even when using spatial acceleration structures (see, e.g., [10]), this can still be considered as too costly for dynamic scenes with a lot of objects.

In interactive rendering, a popular approach is the use of *shadow maps* introduced by Williams [17]. In this two-pass approach, a depth texture is rendered from the position of the light source. In the second pass, the scene is rendered from the camera's position. To check for occlusion, each fragment is transformed to the light source's image space and its depth value is compared to the value stored in the depth texture. If the stored depth value is smaller, the point is shadowed. While conceptually simple and efficient, this approach suffers from artifacts. The resolution of the shadow map directly limits the accuracy of the shadows. In case of spotlights, the shadow map has to be distorted perspectively, leading to artifacts for shadows on objects that are distant from the light source.

To reduce the artifacts, Stamminger et al. [12] proposed to compute the shadow map in the perspectively transformed *camera space*, enhancing the shadow resolution of close objects. To avoid the handling of special cases, Wimmer et al. [18] modified this approach to employ the perspective transformation into *light space*. By modify-

*e-mail: michael.krone@visus.uni-stuttgart.de
†e-mail: guido.reina@visus.uni-stuttgart.de
‡e-mail: ertl@vis.uni-stuttgart.de

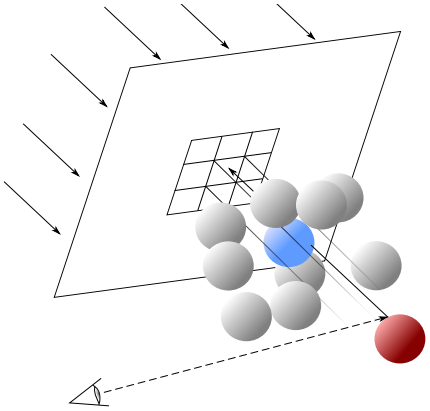

Figure 2: The shadow map is shown here as the large rectangle. Directional light is cast towards the data set. To determine if a point on the red sphere is shadowed, a ray/sphere intersection is computed with the blue sphere, since the parameters of this sphere are stored in the corresponding texel. The ray is cast from the red sphere surface point with inverse light direction.

ing the position of the near clip plane of this transformation, users are also able to control the shadow quality distribution. Zhang et al. [20] suggested to use multiple shadow maps to increase accuracy. The camera's viewing frustum is split into multiple parts, and a separate shadow map is computed for each of them.

Fernando [5] proposed *percentage-closer soft shadows (PCSS)*, which approximates soft shadows resulting from area light sources. The first step of this approach is to sample an area of the shadow map. During this step, the average distance of samples occluding the currently rendered fragment is computed to estimate the width of the penumbra. Afterwards, *percentage-closer filtering* [2] is applied by averaging multiple shadow map samples. The kernel defining the shadow map's sampled area is sized proportional to the approximated penumbra. Distant occluders lead to larger kernel size and, therefore, smoothly blurred shadows.

Removing the typical aliasing artifacts caused by the limited shadow map resolution would require to intersect each occluding triangle. Thus, Story [13] proposed *hybrid ray traced shadows* (HRTS), which uses lists of occluding triangles associated with each shadow map texel instead of single depth values. After transforming a world space point to shadow map coordinates, all triangles stored in a texel can be tested for intersection with the shadow ray, and, therefore, occlusion of the point. The creation of the triangle list shadow map requires *conservative rasterization* [7] offered by Nvidia's Maxwell-generation GPUs. Conservative rasterization makes sure that a fragment is generated even if a triangle only partially covers it. The approach also allows approximation of soft shadows by interpolating between the result and PCSS, for example depending on the distance between shadow ray origin and occluder. *Shadow volumes* by Crow [3] also enable the rendering of perfectly sharp-edged shadows cast by polygons. For this, volumes that enclose the shadowed space are created. To test if a fragment is inside such a volume, the stencil buffer can be utilized [9].

In our case, data sets consist of a large number of spheres implicitly defined by a position and a radius, illuminated by a directional light source. The state-of-the-art method to render large numbers of spheres is GPU-based ray casting [4]. Here, only the implicit description of each sphere is uploaded to the GPU and a pixel-correct representation is computed in the fragment shader via ray-sphere intersections. The aforementioned shadowing techniques would require tessellation of the spheres or sampling depth values to a texture, introducing artifacts. Therefore, we combine shadow maps with implicit representations and ray casting.

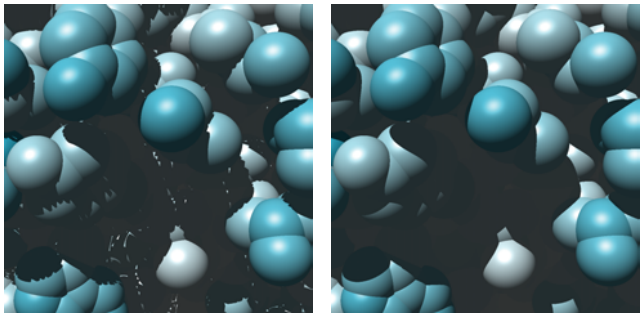

Figure 3: If only one texel is sampled, visible artifacts can occur at the shadow borders (left). Sampling one additional ring of texels around the initial texel removes these artifacts (left).

3 Basic Method

Our method consists of two rendering passes. In the first pass we render an *Implicit Sphere Shadow Map* (ISSM) into a texture: Be l the direction of the directional light source and c the center of the data set's bounding box in world space. We define a camera looking at c that is positioned at $c-(l\cdot u)$, where u is a scalar that is chosen such that the camera position is outside the bounding box of the data set to make sure that all occluders are captured. We then render the data set with an orthogonal projection to achieve parallel rays. Note that due to orthogonal projection, our method is invariant to the distance of the shadow map camera to the scene.

A sphere is implicitly defined by $r=|(x,y,z)-p|$, with r being the sphere radius and p being the sphere position. Instead of the depth of the fragment, p and r of the rendered spheres are written to the texture. The regular depth value is still used for the depth test. Consequently, after the first rendering pass, each texel of the texture contains the parameters of the sphere closest to the image plane.

In the second rendering pass, the scene is rendered using the viewer's camera parameters. It is now possible to check if any fragment is occluded by a sphere by transforming this fragment into the light coordinate system. As in the traditional shadow map approach, the texel of the implicit shadow map corresponding to this fragment is determined. To check for occlusion, a ray can be cast from this point with direction $-l$ and checked for a possible intersection with the sphere stored in the texel (see Fig. 2). It is, thus, possible to find the occluding sphere without explicitly intersecting every sphere in the scene. The main advantage is the use of the analytic intersection with the shadowing sphere. Thus, our approach does not suffer from quantization artifacts.

3.1 Improvements

Similar to regular, depth-based shadow maps, a naïve implementation of our ISSM algorithm requires a relatively large shadow map resolution in order to produce good results. Otherwise, artifacts may appear along shadow edges and even inside shadowed areas. This is an inherent limitation of our method that can have two reasons: 1.) if two spheres partially cover a texel, only the one closer to the light can be stored; 2.) if the sphere does not cover the center of the fragment, it will not be hit by the ray. Consequently, artifacts will occur due to these missing spheres. An example can be seen in Fig. 3 (left). In theory, a simple solution would be to increase the shadow map resolution until these artifacts become invisible.

A solution that does not require high resolutions is to sample a patch of size $(2t+1)^2$ centered on the relevant shadow map texel, with t being the number of texel rings around the central texel. For the shadow computation, every sphere within this area is intersected with the shadow ray. If a sphere covering a texel was overwritten (or not hit) during the shadow map generation pass but is stored within this area, it will be intersected. Fig. 3 shows how sampling a patch containing only one additional ring of texels leads to artifact-free

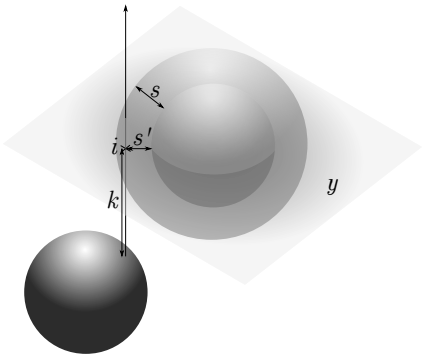

Figure 4: Depiction of the measures used to compute the shadow intensity of soft shadows. The shadow ray is spawned from the lower black sphere. The ray does not hit the sphere above, but will pass through the sphere's extension, hence a partial shadow is cast.

shadows. Our ISSM algorithm produces hard shadows at interactive frame rates even for large data sets, as shown in Sect. 4.1.

3.2 Soft Shadows

Similar to HRTS [13], we could use PCSS [5] to blend the hard shadows with PCSS shadows to obtain soft shadows. Alternatively, it is possible to visually enhance the hard shadows created by our method so that they appear to have soft edges at practically no additional computational cost as schematically shown in Fig. 4. The radii of spheres are virtually increased by a user-defined value s. If a sphere is not intersected by the shadow ray, the distance s' of the ray to the sphere surface is computed. For this, we define the plane y that contains the sphere center and has the shadow ray direction as its normal. Next, the intersection point i of y with the shadow ray is computed. The distance of the ray to the sphere surface s' is the distance from i to the sphere center minus radius r. k is the distance from the shadow ray origin to the intersection i. If s' is smaller than s, the sphere casts a partial shadow (penumbra), whose strength is weighted by the distance to the surface. This value is normalized by s to $s'_n = \frac{s'}{s}$, so $(1-s'_n)^x$ is the shadow's strength depending on the distance of the ray to the sphere's surface. By changing the exponent x applied to $1-s'_n$, users are able to introduce non-linearity to the shadow intensity function. t must be chosen large enough such that the maximum possible penumbra can be captured.

Real-world shadows are sharper the closer the partially occluding object and the occluded surface are. To approximate this, s is scaled by a weighting function depending on k. A user-controlled value b describes the distance until shadows have maximum smoothness. We found a Gaussian weighting produces realistic-looking results approximating an area light. When using linear weighting, the linearity becomes apparent for small b.

It is possible that multiple spheres are partially or completely shadowing a surface point. To account for this case, the applied illumination is minimized among all computed values. Obviously this approach is not physically correct, but convincing results are obtained, as shown in Fig. 5.

The final lighting computation was implemented as follows: If a fragment is completely shadowed, it is only illuminated by the light source's ambient component. In case of partial shadowing resulting from soft shadows, the diffuse and specular component are weighted by the resulting intensity. To avoid particle self-shadowing, occluder and occludee are checked for identity (via their positions).

4 Results and Discussion

The shadows rendered by this approach can add depth cues to the visualization of a data set. However, these cues are not present in unshadowed areas (e.g., a cavity that is directly lit be the light source). The same is true for completely shadowed areas that are only lit by the ambient component of the Phong light. Combining ISSMs with Ambient Occlusion (AO) remedies this issue, since it modulates the ambient term (see Fig. 1). We use the real-time particle AO method presented by Staib et al. [11] in our implementation. Combining AO and ISSM leads to greatly enhanced perception of shape and structure, as shown in Fig. 1 and Fig. 7. While classical ray tracing still produces slightly more realistic results (see Fig. 6), our rendering obtains interactive frame rates, as shown in Sect. 4.1.

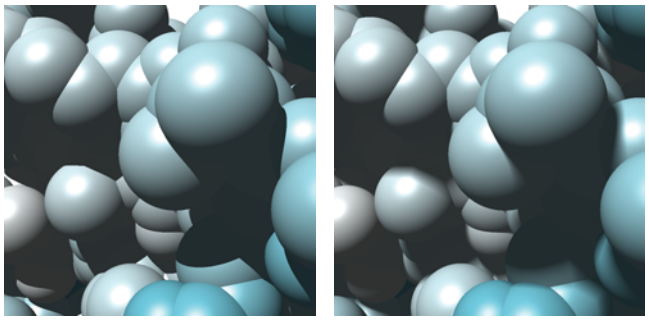

Figure 5: Left: Default shadows with hard edges. Right: Same light direction and camera position, but with soft shadows enabled.

However, it is possible that entire spheres are not casting shadows if the resolution is set too low relative to the data set size, sphere density, and radius (e.g., particles are smaller than the distance between two fragment centers during ISSM creation and can be missed entirely). This can be alleviated by increasing t only if the sphere is stored in the shadow map at all. Otherwise, the shadow map resolution has to be increased. To find an appropriate t, we implemented a procedure that automatically increments t until no changes in the rendered image are detected anymore. However, if the shadow map resolution is too low, this can result in a relatively large sample patch size, but with diminishing returns in image quality for later iterations. Therefore, a tradeoff between shadow map resolution and t has to be found. In practice, we found $t = 1$ to be sufficient when a reasonable shadow map resolution was used.

Note that our approach is closely related to HRTS [13]. However, only one primitive (the closest intersecting sphere) is stored per shadow map texel instead of multiple triangles of tessellated surfaces. Storing a list of d spheres would be possible, but would lead to a decreased frame rate (especially due to the generation of a per-fragment list of primitives [19]). In this case, however, our method could be implemented without hardware-dependent conservative rasterization by slightly increasing the sphere radii when generating the shadow map. Using lists of spheres would remedy the need to sample a patch of texels. However, rendering times would likely not be improved, since usually a patch with $t = 1$ (i.e., 9 texels) is sufficiently large. In contrast, the sphere list generated by HRTS would on average be much larger than that. This is backed by the fact that the HRTS method is not recommended for large scenes [13].

4.1 Performance

We measured the performance of our approach using two different test systems: an Intel Core i7-4770 (4×3.4GHz) with an Nvidia GeForce GTX 760 (2 GB VRAM) as well as an Intel Core i7-3770 (4×3.5GHz) with an Nvidia GeForce GTX Titan (6144 MB VRAM). The viewport resolution was set to 1920×1200 pixels and the same parameters were used for all tests. A shadow map size of 512×512 texels was used for all tests. We used data sets of different sizes from the Protein Data Bank [1] (1TII: 5,000 particles; 1AF6: 10,000; 1AON: 60,000; 1SVA*: 1,000,000) and from Molecular Dynamics simulations of our project partners (NiAl: 85,000 particles). The results are shown in Fig. 8. Note that we also give the frame rate for the combination of ISSM and AO. As mentioned above, we

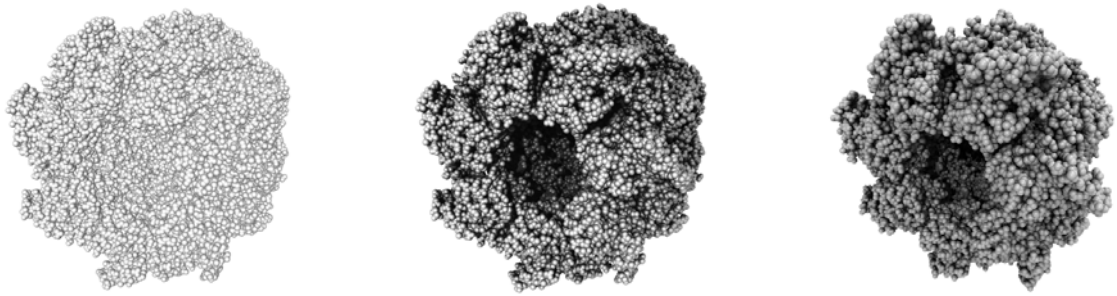

Figure 6: A comparison between local lighting (left), a combination of our ISSM shadows and AO [11] (center), and an image create by the Tachyon ray tracer included in VMD [8] (right). The data set (1AON) is a protein from the PDB [1]. Note how the large cavity in the center of the protein is nearly invisible when using only local lighting. In contrast, the combination of ISSM and AO highlights the structure clearly and achieves similar image quality to the offline ray tracing, which takes several seconds to compute.

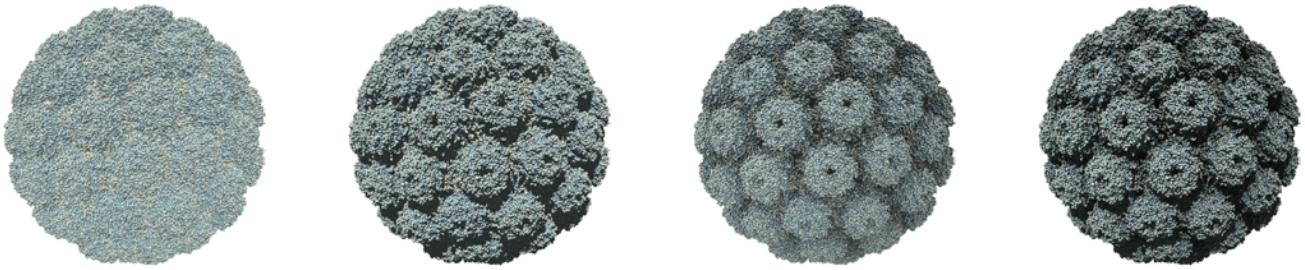

Figure 7: A comparison of lighting methods applied to the largest test data set 1SVA*, a virus capsid of approximately one million atoms, which was instantiated from a single building block (PDB ID: 1SVA). Ambient Occlusion (AO) alone, which is often used in scientific visualization, highlights small-scale structures within the data very well, but lacks a relation to a specific, directional light source. Adding the directional shadows from the ISSM to the AO rendering not only compensates the over-darkening in the top left area but also creates an image that supports depth perception even better, since it is physically more meaningful.

implemented the interactive AO method by Staib et al. [11] and used 16 directions with a maximum distance of 0.7 (in terms of bounding box size) to sample the AO volume. For more details about the AO, please refer to the original publication.

As observable in Fig. 8, our ISSM method scales well with the size of the data set. On our weaker system, we observed around 80% of the performance compared to standard local lighting for all data sets. Our stronger system exhibited similar behavior, with a performance of 73% for our smallest data set and a performance of 80% for our largest one. The reason for this constant impact on the performance is that the computation of the ISSM depends linearly on the size of the data set, since each particle has to be rendered to generate the shadow map texture. For the weaker system and the largest data set of 1 M particles (1SVA*), we still reach an interactive frame rate of 33 fps when using only ISSM, and 31 fps when using ISSM combined with AO. Note that the NiAl data set has a higher performance than 1AON, although it contains 41% more particles. This can be explained by the spatial arrangement of the data, which results in much fewer generated fragments for the spheres.

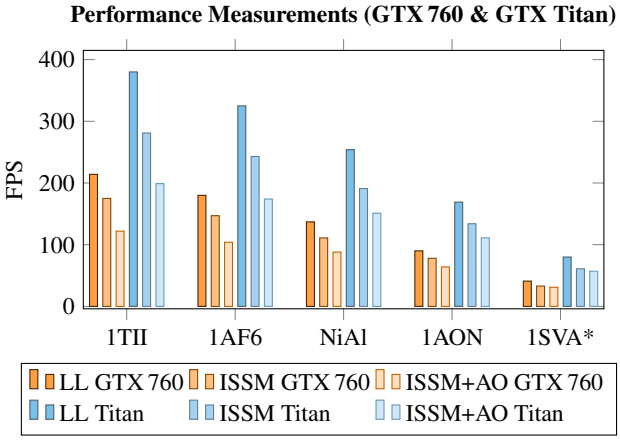

Figure 8: Performance of our ISSM compared to deferred rendering with local lighting (LL, Phong shading) on our two test systems. The palest bar shows the frame rates for ISSM combined with AO.

5 Conclusions and Future Work

We presented Implicit Sphere Shadow Maps (ISSM), an application-tailored variant of shadow maps for large particle data sets. The resulting soft shadows improve the perception of shape and structure. Our ISSM method is similar to HRTS [13], since the shadow map stores the implicit description of the spheres that are hit first when casting directional light rays towards the data. However, only one sphere is stored per texel instead of multiple triangles as with HRTS. Conversely, our implicit representation of whole objects does not require elaborate data structures like per-pixel linked lists. During shading, the texel in the shadow map is determined as in the traditional approach, but the sphere stored in the texel is intersected with a shadow ray, resulting in hard shadows without the typical texel-aliasing of regular shadow maps. The approach maintains interactive frame rates even for large data sets and leaves enough headroom to be combined with interactive Ambient Occlusion.

For future work, we want to extend our ISSM method to spotlights, which should be conceptually straightforwardly. The optimal parameters for this case, however, have to be determined yet.

Acknowledgments

This work was partially funded by Deutsche Forschungsgemeinschaft (DFG) as part of SFB 716.

REFERENCES

[1] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, 2000. doi: 10.1093/nar/28.1.235

[2] M. Bunnell and F. Pellacini. Efficient shadow volume rendering. In *GPU Gems*. Pearson Education, Inc., 2004. Accessed: 24. May 2016.

[3] F. C. Crow. Shadow algorithms for computer graphics. *SIGGRAPH Computer Graphics*, 11(2):242–248, 1977. doi: 10.1145/965141.563901

[4] M. Falk, S. Grottel, M. Krone, and G. Reina. *Interactive GPU-based Visualization of Large Dynamic Particle Data*. Morgan & Claypool Publishers, San Rafael, CA, 2016. doi: 10.2200/S00731ED1V01Y201608VIS008

[5] R. Fernando. Percentage-closer soft shadows. In *ACM SIGGRAPH 2005 Sketches*, SIGGRAPH '05. ACM, New York, NY, USA, 2005.

[6] A. S. Glassner. An introduction to ray tracing, 1989.

[7] J. Hasselgren, T. Akenine-Möller, and L. Ohlsson. Conservative rasterization. In *GPU Gems 2*. Addison-Wesley Professional, 2005. Accessed: 24. May 2016.

[8] W. Humphrey, A. Dalke, and K. Schulten. VMD visual molecular dynamics. *J. Mol. Graph.*, 14:33–38, 1996.

[9] M. McGuire. Efficient shadow volume rendering. In *GPU Gems*. Pearson Education, Inc., 2004. Accessed: 25. January 2016.

[10] J. Revelles, C. Urea, and M. Lastra. An efficient parametric algorithm for octree traversal. In *Journal of WSCG*, pp. 212–219, 2000.

[11] J. Staib, S. Grottel, and S. Gumhold. Visualization of Particle-based Data with Transparency and Ambient Occlusion. *Computer Graphics Forum*, 34(3):151–160, 2015. doi: 10.1111/cgf.12627

[12] M. Stamminger and G. Drettakis. Perspective shadow maps. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pp. 557–562. ACM, 2002. doi: 10.1145/566570.566616

[13] J. Story. Hybrid ray traced shadows. Game Developer Conference, 2015. `https://developer.nvidia.com/content/hybrid-ray-traced-shadows`.

[14] J. Story. Advanced Geometrically Correct Shadows for Modern Game Engines. Game Developer Conference, 2016. `http://developer.download.nvidia.com/gameworks/events/GDC2016/jstory_hfts.pdf`.

[15] M. Tarini, P. Cignoni, and C. Montani. Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Trans. Vis. Comput. Graphics*, 12(5):1237–1244, 2006. doi: 10.1109/TVCG.2006.115

[16] T. Whitted. An improved illumination model for shaded display. *Commun. ACM*, 23(6):343–349, 1980.

[17] L. Williams. Casting curved shadows on curved surfaces. In *Proceedings of the 5th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '78, pp. 270–274. ACM, 1978. doi: 10.1145/800248.807402

[18] M. Wimmer, D. Scherzer, and W. Purgathofer. Light space perspective shadow maps. In *Proceedings of the Fifteenth Eurographics Conference on Rendering Techniques*, EGSR'04, pp. 143–151. Eurographics Association, 2004. doi: 10.2312/EGWR/EGSR04/143-151

[19] J. C. Yang, J. Hensley, H. Grn, and N. Thibieroz. Real-Time Concurrent Linked List Construction on the GPU. *Computer Graphics Forum*, 29(4):1297–1304, 2010. doi: 10.1111/j.1467-8659.2010.01725.x

[20] F. Zhang, H. Sun, L. Xu, and L. K. Lun. Parallel-split shadow maps for large-scale virtual environments. In *Proceedings of the 2006 ACM International Conference on Virtual Reality Continuum and Its Applications*, VRCIA '06, pp. 311–318. ACM, 2006. doi: 10.1145/1128923.1128975