PROCEEDINGS
13th INTERNATIONAL CONFERENCE ON GEOMETRY AND GRAPHICS
August 4-8, 2008, Dresden (Germany)
ISBN: 978-3-86780-042-6

GEOMETRY OF ARBITRARY LIGHT DISTRIBUTIONS

Günter WALLNER

University of Applied Arts, Vienna

ABSTRACT: Sophisticated simulation of lighting is crucial for plausible computer image generation. In many real time applications the simulation of light is restricted to directional, point and spot light sources. This paper presents a method to simulate hemispherical and omnidirectional light sources with arbitrary light distributions by means of light distribution textures (LDTs). These textures can be gained from photometric data files provided by manufacturers of luminaires. The derivation of LDTs from such files is presented. Reconstruction of the photometric solid from the LDT with bilinear and bicubic filtering is also discussed.

Keywords: Lighting, Goniometric Light Sources, Bicubic Interpolation, Real Time Graphics

1 INTRODUCTION

Realistic simulation of lighting is necessary for plausible computer image generation. Many physical plausible rendering systems, like Lightscape, Lightworks, or Radiance, nowadays allow the inclusion of goniometric light sources[1] which permits the accurate simulation of real world luminaires. As pointed out by Albin et al. [1] such simulations are needed in architectural design of galleries, offices (certain lighting conditions have to be met by law) tunnels and so on. Furthermore the lighting industry relies on the accurate simulation of illumination caused by a luminaire before production of the luminaire actually starts.

However, in real time rendering direct illumination is still usually performed with point-, directional- and spot lights and, as it seems, not much work has been carried out in this area. In this paper we show the application of light distribution textures – a concept we introduced in [15] in regard to radiosity – to real time rendering.

The remainder of this paper is structured as follows. Section 2 introduces the necessary concepts and definitions. Related work in the field of computer graphics is reviewed in Section 3. Light distribution textures are covered in Section 4. Implementation details are provided in Section 5 and Section 6 presents results achieved with our method. Finally the paper is concluded in Section 7.

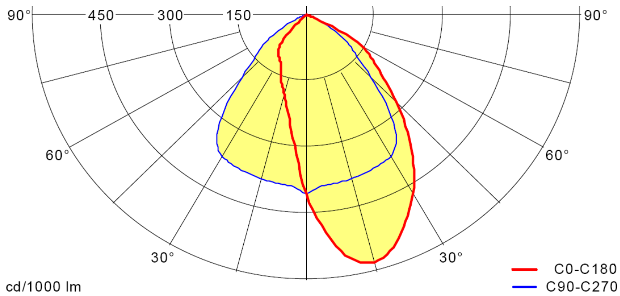

Figure 1: Goniometric Diagram of a asymmetrical spotlight

2 CONCEPTS AND DEFINITIONS

In *far-field* photometry a light source is regarded as a point source for which the inverse square law applies. The inverse square law states that the intensity of light radiating from a point source is inversely proportional to the square of the distance from the source. According to Ashdown [5] this assumption holds true for most architectural luminaires if the distance from the luminaire to the measurement point is at least five times the maximum width of the luminaire (five-times rule). Its photometric distribution is usually expressed as a *goniometric diagram* (see Figure 1). These diagrams represent a planar slice through

[1] A goniometric light source is one which can emit widely varying amounts of light energy in different directions.

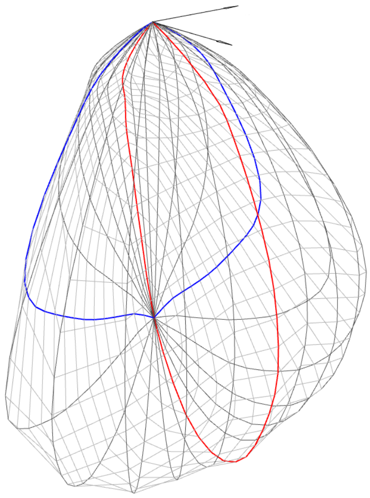

Figure 2: Photometric solid of the asymmetrical spotlight. The slices depicted in the goniometric diagram of Figure 1 are colored red and blue respectively.

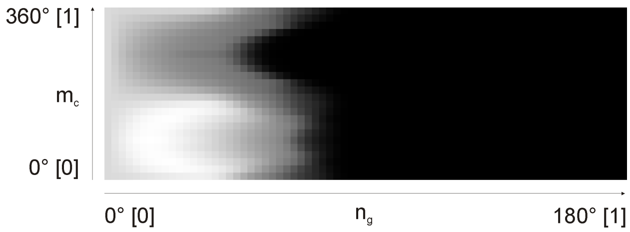

Figure 3: Light distribution texture of the luminaire from Figure 2. Texture coordinates are given in parenthesis.

the light distribution and therefore plot the relative intensity as a function of vertical angle specified in candela. A typical goniometric diagram depicts two perpendicular slices through the intensity distribution in polar coordinates, as noted by Cohen and Wallace [8]. Languénou and Tellier [11] suggested a method for interpolating smoothly between those two slices. However, for luminaires which exhibit a more complex light intensity distribution (LID) further slices are necessary.

The 3-dimensional extension of a goniometric diagram is referred to as *photometric solid* (see [4]). Such a photometric solid depicts the variation over vertical and horizontal angles simultaneously (refer to Figure 2).

The measurements of a source's LID are provided by the manufacturer in terms of a photometric file. Various photometric file formats exist, where – according to Ashdown [5] – EULUMDAT is the de facto industry standard in Europe, while IESNA LM-63 is used by North American lighting manufacturers. We will focus on EULUMDAT files in this paper, though LDT can also be gained from other photometric files. Unfortunately it seems that no official specification on the EULUMDAT standard is available on the web. However, an English translation can be found at [2]. In the remainder of this paper we will use the nomenclature of the EULUMDAT specification where m_c is the number of C-Planes (vertical slices) and n_g is the number of measurements per C-Plane.

3 RELATED WORK

Goniometric diagrams where first introduced to computer graphics by Verbeck and Greenberg [14]. As already pointed out briefly in Section 2, Languénou and Tellier [11] showed how missing values can be interpolated from two perpendicular slices of the goniometric diagram, by projecting the direction vector onto the two planes and then performing an elliptic interpolation between the two retrieved intensity values. Zotti et al. [16] presented a method to approximate a luminaire with a combination of at most two OpenGL lights. Their method did not exploit programmable hardware, only OpenGL's built in light sources and therefore had shortcomings with lights that do not have spotlight characteristics. For global illumination renderer more work on this topic is available. For example, Albin and Peroche [1] proposed a method to reconstruct the light distribution from a goniometric diagram. To accomplish this, a locally supported kernel is associated with every measurement to avoid some problems with bilinear interpolation. They also cite more work in regard to global illumination.

4 LIGHT DISTRIBUTION TEXTURES

In this section we will first explain how an LDT can be derived from a light intensity distribution and then describe the necessary calculations to access such a texture.

An LDT stores the light distribution of a luminaire and has dimension $n_g \times m_c$. The intensity values are read from the photometric file and normalized to the range $[0..1]$ by dividing through the maximum intensity I_{max}. Storing normalized values allows to control the intensity of a light source independently (e.g. for dimming). Figure 3 shows the LDT from the asymmetrical spotlight from Figure 2. In case of a rotationally symmetric light distribution only a one-dimensional texture has to be used. If the light distribution covers just one hemisphere, only the values from that hemisphere have to be stored.

To access the LDT to retrieve the light intensity for a given direction $\mathbf{d}$, the appropriate texture coordinates have to be calculated. For a rotationally symmetric LID and a light source with reference system $(\mathbf{n}_0, \mathbf{r}_0, \mathbf{u}_0)^2$ the angle θ between $\mathbf{d}$ and $\mathbf{n}_0$ has to be calculated and mapped to the range $[0..1]$. This is done by dividing θ by π for a omnidirectional light source or by $\pi/2$ for a hemispherical source. Consequently the texture coordinates are given by

$$\begin{aligned} s &= \frac{\arccos(\mathbf{n}_0 \cdot \mathbf{d})}{\pi} \\ t &= 0.5 \end{aligned} \tag{1}$$

For a luminaire with an asymmetric luminous flux distribution the issue is a little bit catchier, because the luminaire has to be orientated correctly in the scene. This is problematic, because – as noted by Ashdown [5] – various IESNA LM-series documents provide contradictory specifications on how the photometric solid is to be orientated with respect to the physical outline of a luminaire. See [5] for a more in-depth discussion on this topic. To obtain the correct C-plane for a non-rotationally symmetric LID, $\mathbf{d}_0$ is projected orthogonally onto the plane $(L, \mathbf{u}_0, \mathbf{r}_0)$. Afterwards the angle between the projected vector and $\mathbf{r}_0$ is calculated. Finally, we have to evaluate if $\mathbf{d}$ is in the positive or negative halfspace in respect to the plane $(L, \mathbf{n}_0, \mathbf{r}_0)$. The t-coordinate is therefore given by

$$t = \begin{cases} 1 - \frac{0.5\varphi}{\pi} & u_0 \cdot \mathbf{d} \leq 0 \\ \frac{0.5\varphi}{\pi} & \text{otherwise} \end{cases} \tag{2}$$

[2] The subscript 0 is used to denote unit vectors.

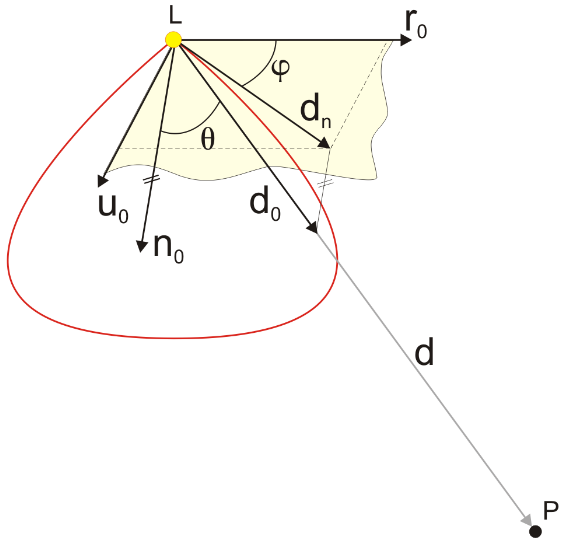

Figure 4: Geometrical relationships which are necessary to sample the LDT for a given light source L and a point P. The light distribution is shown as red curve.

where

$$\varphi = \arccos(\mathbf{r}_0 \cdot ((\mathbf{d} \cdot \mathbf{r}_0)\mathbf{r}_0 + (\mathbf{d} \cdot \mathbf{u}_0)\mathbf{u}_0)) \tag{3}$$

The geometric situation is illustrated in Figure 4. By setting the interpolation mode of the texture to linear, missing values are automatically interpolated by graphics hardware. The texture wrap mode in s-direction is set to *GL_REPEAT* to ensure continuity between 0° and 360°.

4.1 Filtering

Mapping the light distribution to a two-dimensional texture has the advantage that two-dimensional filtering methods can be used to interpolate missing values.

Ashdown [3] points out that simple bilinear interpolation between the nearest measurement angles is probably adequate for all practical applications. However, in cases where only a small number of measurements is provided for a luminaire, interpolation artifacts may appear, since it gives only a piecewise linear approximation. Further problems with bilinear interpolation arise if the measurements are not taken regularly, as noted by Albin and Péroche [1]. In such cases higher-order interpolation schemes can be used to improve the sampling of the LDT. Bicubic filtering yields itself well to the task because (a) the interpolated result is smooth in all directions and (b) it can be performed very fast on programmable graphics hardware. For instance, Bjorke [6] presented a method to perform bicubic filtering with the Mitchell-Netravali kernel in the fragment shader. Sigg and Hadwiger [13] presented a method to perform cubic b-spline filtering with linear texture fetches instead of repeated nearest-neighbor sampling (as done by Bjorke). This reduces the number of required texture samples and therefore increases the performance even further. Note that in case of LDTs an interpolating filter is required, since the provided data is certain to be part of the photometric solid[3]. Results for reconstructing the photometric solid with Catmull-Rom splines are provided in Section 6.

5 IMPLEMENTATION

The presented method can be easily included into existing surface reflection shaders (like e.g. Blinn-Phong reflection model) by replacing the constant light intensity with the intensity sampled from the LDT. The light intensity used for shading a surface point is therefore given by

$$B = I \cdot LDT(\theta, \varphi) \cdot \frac{1}{d^2} \tag{4}$$

where I is the absolute intensity of the light source, $LDT(\theta, \varphi)$ is the normalized intensity sampled from the LDT and $1/d^2$ accounts for the inverse square law of light (d being the distance from the lightsource). The sourcecode for accessing the LDT in a fragment shader is given in Listing 1.

We implemented the dual paraboloid shadow mapping method of Brabec et al. (refer to [7]) to show results in conjunction with shadows. It should be noted that any other shadowing technique which supports omnidirectional or hemispherical light sources, like Gerasimovs omnidirectional shadow mapping [10] or Shadow Volumes [9], can be used too.

[3]This is not the case for the cubic b-spline filtering method presented in [13]. However, the method can be adopted to e.g. Catmull-Rom splines.

```
float3 d = normalize(lightPosition - pos);

// calculate s-coordinate
float cosi = -dot(lightNormal, d);
float phii = -min(acos(cosi) - PI, 0) / PI;

// calculate t-coordinate
float3 up = cross(lightNormal, lightRight);
float3 dn =
    normalize(dot(-d,lightRight)*lightRight +
    dot(-d,up)*up);
float cosr = dot(lightRight, dn);
float W = 0.5*(acos(cosr)/PI);

float cosu = -dot(up, d);
float f = max(0, sign(cosu));
float texV = (1-W)*f + W*(1-f);

// sample light intensity from the LDT
lightIntensity = tex2D(LDT,
    float2(1-phii,texV));
```

Listing 1: Cg fragment shader code for accessing a non-rotationally symmetric omnidirectional light distribution texture

A floating point off-screen buffer is used to allow high dynamic range lighting. On this buffer a tone mapping operation is performed to map the values to the displayable output range [0..1]. For the images in this paper we used a simple exposure function defined by $y = 1 - e^x$.

6 RESULTS

Figure 5 shows a scene which is lit by a circular high-bay luminaire which exhibits a *winged* emission pattern. Disco and effect lights can also be simulated with LDTs (see Figure 7). It is worth mentioning that similar effects can be reached with projective texture mapping (see [12]). However, in case of projective texture mapping the texture is only projected onto the surface, it does not influence the lighting calculation. Furthermore it is not possible to project the texture onto the whole halfspace at once. Figure 6 shows a room lit by four light sources. For the luminaire at the ceiling a compact downlight with extremely wide light distribution was used. For the lamp on the wall rotationally symmetric spotlights with a slightly anomalous distribution were used, and an asymmetrical light distribution of a downlight was assigned to the desk lamp. Average rendering times over 300 frames for different scenes on an Intel Core2 with 2.13 GHz, 3.5

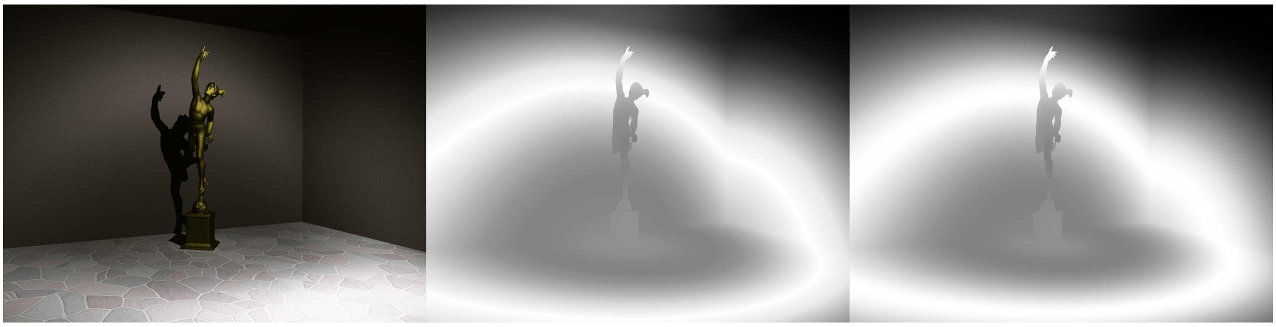

Figure 5: Roman god mercury lit by a rotationally symmetric high-bay luminaire (left). The bilinear sampled light intensity from the LDT which is used for shading a surface point (middle) compared with the result of bicubic interpolation (right). White means maximum intensity whereas black is zero intensity. The goniometric diagram of the luminaire is depicted in Figure 8.

Table 1: Average rendering times

n_t	n_l	$t_s[ms]$	$t_g[ms]$	$inc[\%]$
96	1	2.106	2.161	2.61
	4	4.446	4.692	5.16
	7	7.300	7.465	2.26
1760	1	1.603	1.686	5.18
	4	3.423	3.729	8.94
	7	4.781	5.374	12.4
14764	1	2.292	2.473	7.9
	4	5.031	5.619	11.69
	7	7.906	8.863	12.1
114445	1	4.435	4.588	3.45
	4	10.051	10.322	2.7
	7	16.601	17.261	3.98

GB RAM and a Geforce 8800GTS with 640MB RAM are given in Table 1. The table states the number of triangles n_t in the scene as well as the number of light sources n_l. The average rendering time of a frame if standard point light sources are used is denoted as t_s and the average rendering time with asymmetrical light distributions by means of bilinear interpolated LDTs is designated as t_g. The increase in rendering time is given in the column *inc*. In both cases the same render settings[4] were used, only the shader for calculating the lighting was interchanged. As shown, the increase in rendering time is tolerable and can be further reduced if only symmetrical light distributions are used.

To compare the quality of the reconstruction of the photometric solid from an LDT, we imple-

[4]This includes accumulating the contribution of each light source in a frame buffer object and subsequent tone mapping.

Figure 6: A room lit by four goniometric light sources. Note the light distribution at the back wall, which does show a slightly anomalous spot light characteristic.

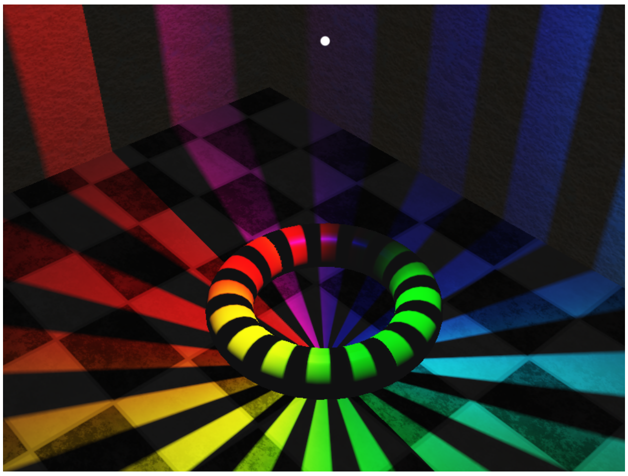

Figure 7: LDTs can also be used to simulate various light effects, like disco lights. The white dot represents the position of the light source.

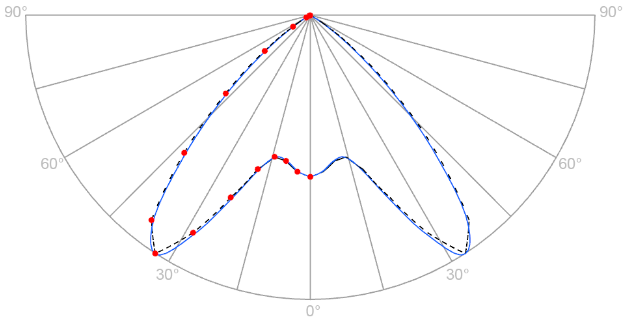

Figure 8: Reconstruction of the photometric solid from the rotationally symmetric light distribution of the luminaire used in Figure 5.

mented Bjorkes [6] bicubic filtering method, because it allowed us to easily try different cubic splines by changing the filter weights in the lookup texture. To compare the results of various interpolation methods, the LDT was rendered to a five-times larger (floating point) texture. Afterward the contents of the texture were obtained and used for geometric reconstruction of the photometric solid. Currently the best results were achieved with Catmull-Rom splines. Figure 8 compares the results for a rotationally symmetric light distribution. Red points depict data present in the photometric data, which were omitted in the right half. The blue line shows bicubic interpolation with Catmull-Rom splines and the black dashed line linear interpolation. Bilinear interpolation works well as long as the intensity does not change much over angular distance. Note, e.g. the bright white stripe in Figure 5 right, which corresponds to the intensity maximum at about 35° in the goniometric diagram. Such artifacts are not visible with bicubic interpolation since the derivatives are continuous over the photometric solid.

7 CONCLUSIONS

In this paper we showed the application of light distribution textures to real time rendering. LDTs allow the fast rendering of hemispherical as well as omnidirectional light sources with directionally dependent light distribution. They can easily be incorporated into existing surface reflection shaders. It should be stressed that in this paper we did not focus on physical plausibility. If this is desired, factors like e.g. the dependency of the luminous flux from the ambient temperature have to be considered. In addition LDTs allow the simulation of various effect lights. Time measurements have shown that the increase in rendering time is acceptable. However, there are still some code optimizations possible, which will result in further performance gains. Finally, different interpolation methods for reconstruction of the photometric solid were discussed. As possible future work the viability of Albin and Péroche's method [1] to real time rendering should be investigated. We are currently experimenting with different kind of bicubic reconstruction filters. To compare the results we are using the Hausdorff Distance Metric, as suggested by [4]. It may be also from interest to compare those results with the ones archived if using the method of Albin and Péroche.

REFERENCES

[1] S. Albin and B. Peroche. Directionally dependent light sources. In *Journal of WSCG*, volume 11. University of West Bohemia, February 2003.

[2] Ian Ashdown. English translation of the EULUMDAT specification. available online: `http://www.helios32.com/Eulumdat.htm`.

[3] Ian Ashdown. Parsing the IESNA LM-63 photometric data file. 1998.

[4] Ian Ashdown. Comparing photometric distributions. *Tech. rep., Department of Computer Science, University of British Columbia*, 1999.

[5] Ian Ashdown. Thinking photometrically part II. *LIGHTFAIR 2001 Pre-Conference Workshop*, March 2001.

[6] Kevin Bjorke. High-quality filtering. In *GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics*. Addison-Wesley Professional, 2004.

[7] Stefan Brabec, Thomas Annen, and Hans-Peter Seidel. Shadow mapping for hemi-

spherical and omnidirectional light sources. In *Advances in Modelling, Animation and Rendering (Proceedings of Computer Graphics International)*, pages 397—408, July 2002.

[8] Michael F. Cohen and John R. Wallace. *Radiosity and Realistic Image Synthesis*. Morgan Kaufmann, 1995.

[9] Cass Everitt and Mark J. Kilgard. Practical and robust stenciled shadow volumes for hardware-accelerated rendering. 2002.

[10] Philipp S. Gerasimov. Omnidirectional shadow mapping. In *GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics*. Addison-Wesley Professional, 2004.

[11] E. Languénou and P. Tellier. Including physical light sources and daylight in global illumination. *Proceedings of the Third Eurographics Workshop on Rendering*, pages 217–225, 1992.

[12] Mark Segal, Carl Korobkin, Rolf van Widenfelt, Jim Foran, and Paul Haeberli. Fast shadows and lighting effects using texture mapping. *SIGGRAPH Comput. Graph.*, 26(2):249–252, 1992.

[13] Christian Sigg and Markus Hadwiger. Fast third-order texture filtering. In *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation*, pages 313–329. Addison-Wesley Professional, 2005.

[14] Channing Verbeck and Donald Greenberg. A comprehensive light-source description for computer graphics. *IEEE Comput. Graph. Appl.*, 4(7):66–75, 1984.

[15] Günter Wallner. GPU radiosity for triangular meshes with support of normal mapping and arbitrary light distributions. In *Journal of WSCG*, volume 16, Plzen-Bory, Czech Republic, February 2008. University of West Bohemia.

[16] Georg Zotti, Attila Neumann, and Werner Purgathofer. Approximating real-world luminaires with opengl lights. In *WSCG 2005 Short Paper Proceedings*, pages 49–52, Plzen, February 2005. University of West Bohemia, UNION press.

ABOUT THE AUTHOR

1. Günter Wallner, Dipl.-Ing., studied computer science at the Vienna University of Technology and is currently a doctoral student at the Department of Geometry at the University of Applied Arts Vienna. His research interest are Computer Graphics, GPU Programming as well as Graph Drawing. He can be reached by e-mail: wallner.guenter@uni-ak.ac.at