VISUAL HULL CONSTRUCTION FROM SEMITRANSPARENT COLOURED SILHOUETTES

J R Rankin and M Boyapati

La Trobe University, Australia

ABSTRACT

This paper attempts to create coloured semi-transparent shadow images that can be projected onto multiple screens simultaneously from different viewpoints. The inputs to this approach are a set of coloured shadow images and view angles, projection information and light configurations for the final projections. We propose a method to convert coloured semi-transparent shadow images to a 3D visual hull. A shadowpix type method is used to incorporate varying ratio RGB values for each voxel. This computes the desired image independently for each viewpoint from an arbitrary angle. An attenuation factor is used to curb the coloured shadow images beyond a certain distance. The end result is a continuous animated image that changes due to the rotated projection of the transparent visual hull.

KEYWORDS

Coloured object reconstruction, inverse problem, visual hull, data driven model, light attenuation factor.

1. INTRODUCTION

Inverse problem research has been undergoing tremendous research in recent years. This involves work particularly done in the area of constructing 3D object models from 2D images, shadows and silhouettes [1-6]. The applications were designed either from an artistic approach or for 3D printing for later use in visual effect computations.

In this paper we present a similar inverse light transport method to compute a 3D visual hull for coloured shadow images. This means we construct a 3D object from different 2D semi-transparent images from multiple viewpoints. Each coloured image is stored in a transparent cube that is made of voxels. It also stores information such as viewpoint projection and direction for input light sources to compute forward rendering images for various viewing angles.

Our paper has the following contributions to this area:

a. A scene model to construct a visual hull for coloured semi-transparent images.
b. A shadowpix type method to incorporate varying ratio RGB values for each voxel. This computes the desired image independently for each viewpoint from an arbitrary angle.
c. An attenuation factor is used to curb the coloured shadow images beyond a certain distance.

DOI : 10.5121/ijcga.2013.3405

2. RELATED RESEARCH

Data driven models are the emerging interest in recent years in the field of computer graphics. They construct the physical objects and manipulate them under different lighting conditions to get the desired effect of the user. Examples include [1], [7] that propose methods to get a desired caustic image using reflection and refractions. Other examples include [8], [9] that design 3D models from real-world physical objects and compute methods to design variations by following a set of key constraints. Another paper with the data driven approach [10] computes shadows, transforms and places them as required by the user by changing the scene geometry. This again doesn't support coloured shadows.

3D modeling of 2D silhouettes using different viewpoints [5], [11] take line drawings as input to generate full 3D coloured objects using CSG techniques. A similar method [12] is used to generate free-form 3D surfaces by placing primitives of 2D image. However this model uses a single image with annotations to compute the final object. The primitives such as cylinders and ellipsoids can be tweaked to get a smooth model. Translucent shadow maps [13] use efficient methods to add shadows to arbitrary scenes. They take into account depth and incident light information and provide optimization techniques to compute sub-surface scattering of translucent objects to derive the shadow maps. However the translucency has uniform wavelength of light with no coloured reflections and transmissions. They are similar to our approach in that they have varying light and material properties although for the same viewpoint.

Visual hull construction for silhouette-based images [14] started in early 90s. This paper creates a black and white volume intersection method based on projections from multiple viewpoint input images. The input images are opaque and the final projected geometry is non-transparent. The shadow art method [2] use different binary images from different viewpoints to create a visual hull. They then tweak and modify the 3D hull using some shadow constraints to get accurate descriptions of the input binary images. Once this process is completed they project the visual hull with point and directional light sources from different points to cast the required shadows in different directions.

A drawback of this approach is that the shadow hull only handles opaque images and coloured semi-transparent images are inconsistent with this way of handling images. In other words, the different semi-transparent coloured voxels when projected onto the screen may get accumulated to different colour pixels rather than the desired image. Our method described in this paper is an extension to this approach. Another method [15] proposes methods to tell whether or not a connected 3D shape can have 3 given rectilinear silhouettes. However it doesn't provide suggestions to change the configuration for the 3D object.

Creating physical objects from coloured shadow images [16] was recently done using a set of stacked transparent materials to form a multi-layer attenuator. The input to this process is the same as our method described in this paper. It takes a number of lighting configurations and desired images to compute the result and print it on a transparent media. When projected with the input lights it displays the required colour shadow images. This paper varies from our approach in that it produces a shadow image of the coloured image using a transparent medium, whereas our method tries to produce the coloured shadow of refracted material such as glass, plastic or marbles.

A recent algorithm called the shadowpix [17] uses different images embedded on a single surface. When lit from different directions, different images are formed due to self shadowing of the surface from those input images. We use this technique to generate multiple coloured shadows from different viewpoints using input lighting configurations. However rather than embedding the images onto a single surface we sculpt them inside a transparent cube so that rotating the cube can create interesting visual patterns.

3.OUR APPROACH

3.1 Setting the stage

The environment consists of required lighting configurations on the left and desired input image on the right. A transparent cube is placed in the middle to capture the target input image. As the input image is changed the cube is rotated to a different angle to capture the new image information.

3.2 Approach

Imagine a transparent cube (which we will refer to as the light cube), which is made up of n^3 smaller cubes with n along each direction. Each of the smaller cubes (we shall call them voxels) has a different colour but is transparent. A beam of parallel rays of white light with a square cross-section shines onto the cube from one side. On the far side of the cube a white screen is placed. This produces interesting colours on the screen. Next the cube is rotated about the vertical axis through its centre and the colour shadows change in appearance in a fascinating way.

3.3 Analysis

We will consider the light cube sitting flat at the centre of a round horizontal table which can rotate but only about the vertical axis through its centre. Therefore the beam of light can be divided into horizontal sheets passing through the light cube to arrive at the different pixel scan lines on the display screen. We can now consider the light of one horizontal section through the cube. Figure 1 below shows a top down view of this two dimensional problem for the simple case of n = 3. The cross-section of the light cube is depicted as the square ABCD.

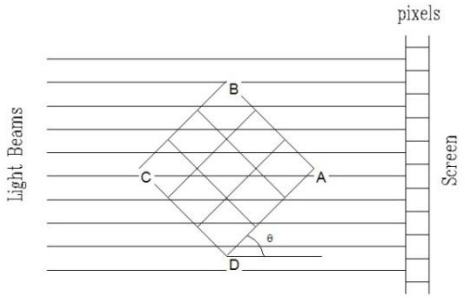

Figure 1. Protopype model of our approach.

Figure 1 above shows parallel light rays from the left striking the screen on the right in a row of pixels. The light source is assumed to be pure white light. Some of the light rays have passed through one or more voxels of the light cube. The light cube has been turned through angle θ

radians (about its centre). The colour of the resulting pixel on the screen is computed based on the length of the sections of the beam in each voxel cell and the colour of the corresponding voxels. Assume that cell i,j has transparency coefficients k_{Rij} for the red component, k_{Gij} for the green component and k_{Bij} for the blue component. It will be more useful to us to use the extinction factors:

$$\alpha_{Rij} = -\ln(k_{Rij})$$
Eq(1a)

$$\alpha_{Gij} = -\ln(k_{Gij})$$
Eq(1b)

$$\alpha_{Bij} = -\ln(k_{Bij})$$
Eq(1c)

Then beam k will produce pixel p_k (of the current scan line) at the screen dependent on which voxels it passes through with colour components p_{Rk}, p_{Gk}, p_{Bk} where

$$p_{Rk} = e^{-\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{Rij}\frac{L_{ijk}}{L}}$$
Eq(2a)

$$p_{Gk} = e^{-\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{Gij}\frac{L_{ijk}}{L}}$$
Eq(2b)

$$p_{Bk} = e^{-\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{Bij}\frac{L_{ijk}}{L}}$$
Eq(2c)

The quantity L_{ijk} is the distance that beam k of the horizontal sheet of light travels through voxel i,j and the quantity L is the maximum horizontal distance through a voxel namely $L = \sqrt{2}s$ where s is the side length of a voxel i.e. $s = S/n$ where S is the side length of the light cube. The colour components have values in [0,1] and those beams k which miss the colour cube have $L_{ijk} = 0$ and therefore produce pure white pixels (1,1,1) on the screen.

3.4 Forward and Backward Computations

The Forward Computation is to compute the pixel value p_k per scan-line given the transparencies k_{Rij}, k_{Gij} and k_{Bij} of each cell on a cross section of the light cube and the rotational angle θ of the light cube. This computation is straight forward to implement though requiring a considerable number of lines of code. As the light cube rotates it shows some fascinating colour variations on the screen. The more complex computation is the Backward Computation. In the Backward Computation we are given the pixel colours $p_k = (p_{Rk}, p_{Gk}, p_{Bk})$ for each scan-line for each of a set of discrete rotation angles θ_m. The number of angles possible (the range of m) is proportional to n. From this data we are to compute the transparencies for all voxels k_{Rij}, k_{Gij} and k_{Bij} for i and j = 1

to n. Using the extinction factors instead of the transmission coefficients the equations become linear as follows:

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{fij}L^{m}{}_{ijk} = -L\ln(p^{m}{}_{fk})$$

Eq(3)

where f takes values of R, G and B. Because of the linearity of these equations we can adapt traditional methods for solving the linear system. Clearly if k ranges from 1 to roughly close to n (but larger) then to have enough equations to solve for the $3n^2$ unknowns α_{fij} we will need close to n rotations of the light cube. This means that the light cube of n^3 voxels can store n predefined graphics images or n frames of a colour animation. Traditional methods will not work directly because the matrix of coefficients $L^{m}{}_{ijk}$ can possibly be singular. Instead we adapt the Gauss-Seidel algorithm to throw out inconsistent equations and undetermined coefficients are set to full transparency. This has been tested for the low count case n = 3 where the inconsistencies are interferences between the images stored in the light cube and they appear as noise in the reproduced images. Because the proportion of inconsistencies is low compared with the number of unknowns we expect that the images rendered will still be very recognizable.

4. RESULTS

We initially tested this approach with a simple 3x3x3 voxel grid for efficiency purposes. The colours in a particular voxel can be a combination of Red, Blue and Green values. Alpha transparency is added into the final coloured voxel to attain transparency. The attenuation factors are computed as per the frequency of the colour of the voxel according to Equ(1). A program was written to modify the standard row reduction method for solving linear systems where the number of unknowns and number of equations are unequal. The algorithm was tested by simulating one layer of a slowly rotating cube with randomly set voxels and produced the physically correct pixelizations per angle of rotation as depicted in Figure 2 below.

When viewed as a 2D screen animation each frame is clearly only of resolution 5x3 which is far too small for recognizable animations. This experimental prototype has however shown that the computational formulation is correct and is solvable in real time for suitably slow rotation of the cube. The Backward computations were not tested. This is left for future work since we had no animation frames to enter. Future research will also increase the voxel size of the colour cube to n = 300 and more for which many more frames will need to be entered (300 in that case). This will therefore considerably increase the Backward computations. However those computations need only be done once. We envisage that the Backward calculations will be done prior and separately from the Forward computations which show the animations as the colour cube rotates. We will also generate the frames by ray-tracing a moving geometric scene and then storing the fames in a file. This process therefore uses three programs. The first program generates a file of frames for an animation by ray-tracing. The second program inputs these frames and computes the voxel colours for the colour cube using the Backward algorithm described in this paper, and stores these voxels in a file. The third program reads in the voxel file data and produces the screen animation via the Forward computation algorithm described in this paper. It will be crucial to find out to what extent the intermediate angles between frames causes interference. It is possible that the interference between frames may make it difficult for the viewer to understand the animation. On

the other hand the interferences may be small as suggested by our calculations or the between frames may be acceptably smooth transitions such that the viewer has no trouble interpreting the animation. This research will be reported in a later paper.

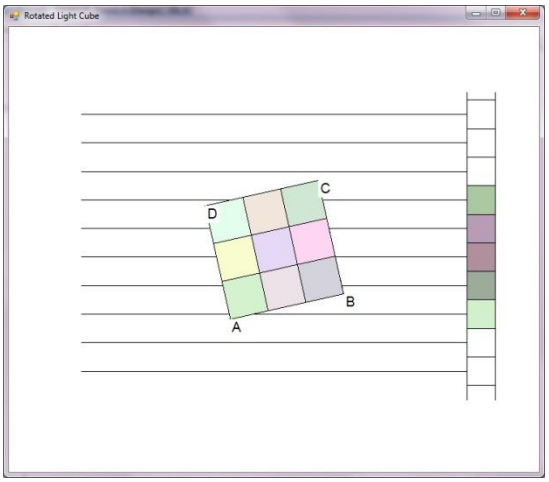

Figure 2. Implementation of the coloured transparent cube.

5. CONCLUSION AND LIMITATIONS

It is easy to ensure that there are more equations than unknowns by over-defining the data with extra input images. The row reduction method can be adapted to throw out inconsistent equations to reach a solution. This however can result in interference in the images stored in the light cube. The resulting reproduced images however appear to show low noise levels. Research is progressing on testing this conjecture further for higher resolutions.

REFERENCES

[1] M. Papas, W. Jarosz, W. Jakob, S. Rusinkiewicz, W. Matusik, and T. Weyrich, "Goal-based Caustics," *Computer Graphics Forum*, vol. 30, no. 2, pp. 503–511, 2011.

[2] N. J. Mitra and M. Pauly, "Shadow Art," *ACM Transactions on Graphics*, vol. 28, no. 5, p. Article 156, 2009.

[3] B. Tandianus, H. Johan, and H. S. Seah, "Caustic Object Construction Based on Multiple Caustic Patterns," *Journal of WSCG*, vol. 20, no. 1, pp. 37–46, 2012.

[4] S. N. Sinha and M. Pollefeys, "Multi-View Reconstruction Using Photo-consistency and Exact Silhouette Constraints: A maximum-Flow Formulation," in *Proceedings of the Tenth IEEE International Conference on Computer Vision*, 2005, pp. 349–356.

[5] A. Rivers, F. Durand, and T. Igarashi, "3D Modelling with Silhouettes," *ACM Transactions on Graphics*, vol. 29, no. 4, p. Article 109, 2010.

[6] M. Finckh, H. Dammertz, and H. P. A. Lensch, "Geometry construction from caustic images," in *Proceedings of the 11th European conference on Computer vision: Part V*, 2010, pp. 464–477.

[7] T. Kiser, M. Eigensatz, M. M. Nguyen, P. Bompas, and M. Pauly, "Architectural Caustics — Controlling Light with Geometry," in *Advances in architectural geometry 2012*, 2012, pp. 91–106.

[8] S. Xin, C.-F. Lai, C.-W. Fu, T.-T. Wong, Y. He, and D. Cohen-or, "Making Burr Puzzles from 3D Models," in *ACM SIGGRAPH 2011 papers on - SIGGRAPH '11*, 2011, no. 1, p. Article no. 97.

[9] M. Hirose, J. Mitani, Y. Kanamori, and Y. Fukui, "An Interactive Design System for Sphericon-Based Geometric Toys Using Conical Voxels," *Lecture Notes in Computer Science*, vol. 6815, pp. 37–47, 2011.

[10] F. Pellacini, P. Tole, and D. P. Greenberg, "A User Interface for Interactive Cinematic Shadow Design," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 563 – 566, 2002.

[11] M. Masry, D. Kang, and H. Lipson, "A freehand sketching interface for progressive construction of 3D objects," *Computers & Graphics*, vol. 29, no. 4, pp. 563–575, Aug. 2005.

[12] Y. Gingold, T. Igarashi, and D. Zorin, "Structured Annotations for 2D-to-3D Modeling," in *ACM SIGGRAPH Asia 2009 papers*, 2009, p. Article No. 148.

[13] C. Dachsbacher and M. Stamminger, "Translucent Shadow Maps," in *Proceedings of the 14th Eurographics workshop on Rendering*, 2003, pp. 197–201.

[14] A. Laurentini, "The visual hull concept for silhouette-based image understanding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 2, pp. 150–162, 1994.

[15] J. Keiren, F. van Walderveen, and A. Wolff, "Constructability of Trip-lets," in *25th European workshop on Computational Geometry*, 2009, pp. 251–254.

[16] I. Baran, P. Keller, D. Bradley, S. Coros, W. Jarosz, D. Nowrouzezahrai, and M. Gross, "Manufacturing Layered Attenuators for Multiple Prescribed Shadow Images," *Computer Graphics Forum*, vol. 31, no. 2pt3, pp. 603–610, May 2012.

[17] A. Bermano, I. Baran, M. Alexa, and W. Matusik, "SHADOWPIX: Multiple Images from Self Shadowing," *Computer Graphics Forum*, vol. 31, no. 2pt3, pp. 593–602, 2012.