Non-photorealistic shape cues for visualisation

Peter Hall*
Department of Computer Science
PO Box 600
Victoria University of Wellington
New Zealand
Email: peter@comp.vuw.ac.nz

Abstract

Visualisation applications often need to display objects that are both three dimensional and coloured. The objects may be represented either by surfaces or by volumes. The motivation behind such displays is to demonstrate two or more independent variables at once. So one variable may determine object shape and another variable may determine the distribution of colour. A lighting model is needed to provides cues for visual shape recognition. Conventional lighting models provide these cues by altering colour, Hence a conflict arises when this approach is adopted because the total colour variation is made up of two components that are perceptually indistinguishable. There are variations that show the distribution of a variable, and global variations caused by the lighting model. It might be said that a side effect of the light model is to distort the information represented by colour.

This paper offers a solution to this problem be replacing the conventional lighting model with a structured lighting model of the kind used in machine vision contexts. The proposed solution simulates the shadow of a regular, planar, grid on the surface of an object. Improved shape cues can be obtained by using a grid that is sensitive to lighting conditions, this generates a cross-hatch shading effect. The notion of a planar grid easily generalises to three dimensions, where it can be used not only with surfaces, but also with volumes. The benefit of the method is that colour is effected only locally, and in well defined places; the remaining colour is readily identifiable and faithfully represents underlying data. In addition, the method in either of its forms is simple to incorporate into existing renderers.

The method is demonstrated with results obtained from a computational fluid dynamics application. Both coloured iso-surfaces and coloured volumes are shown.

1 Introduction

The underlying problem of interest here is that of visualizing how two or more scalar fields are related in the same three-dimensional space. For instance, in computational fluid dynamics (CFD) many fields co-exist in the same space, and the engineer is interested in them all. A collection of visualisation methods such as surface fitting, volume rendering, particle advection and animation may all be used to visualise such complicated data simultaneously[1]. We consider two approaches to this problem: *attribute mapping* and *volume rendering*.

*. This work was performed while the author was a member of staff in the Computer Science Department, Sheffield University, UK.

Under attribute mapping, an iso-surface is fitted to one of the data sets. Techniques for surface fitting are documented in the literature[2,3]. Once fitted, the surface is then coloured to show the distribution of a second field over it. The second field acts as a three-dimensional texture map[4,5]. Typically, the resulting coloured surface is rendered using a conventional lighting model that yields a high fidelity representation of shape. To achieve this surface colour is effected over all the surface. Consequently the integrity of the information represented as colour is compromised. We conclude that if the fidelity of information that is represented by colour is to remain high then conventional lighting models cannot be used. The challenge is to provide a shape cue without affecting surface colour too widely

Volume rendering methods yield images of a data field in three dimensions. Although it was originally conceived for use with scalar fields[6,7], the examples used here demonstrates the method applied to vector fields[8]. In this approach each vector is ascribed a colour that depends upon its direction and magnitude. This is done by placing each vector inside a spherical colour, with the tail of the vector at the origin. The colour pointed to by the head of the vector is used to represent that vector in a visualisation. Here, we note that vector fields cannot be used to define a surface in the manner common the conventional volume renderers, and that every colour in the gamut is used. In this case shape cues that depend on the existence of surfaces cannot be used.

We propose replacing the conventional lighting model with a structured lighting model. The notion of a structured lighting model is not new to researchers in Computer Vision, where it has been found that robots can more easily detect the orientation of surfaces if the shadow of a regular grid is cast upon them [9,10]. The distorted shadow pattern that results gives clues about the shape of the surface. The algorithm presented follows this approach, expect that surface colour is changed only where it lays in the shadow of a grid line. Hence, changes in surface colour are confined to well defined regions that are small compared to the total surface area. Also, we abstract structured lighting into three dimensions, making it useful in a contexts where no surface information is available. Here, the grid comprises a set of mutually orthogonal planes.

If surface information is available then a second generalisation is sufficient to improve shape cues by simulating cross-hatch shading for both two dimensional and three dimensional grids. Cross-hatching shading is a technique often used by artists, especially when drawing with a fine-point stylus such as a pen. Under cross-hatching more and/or thicker lines are drawn in darker regions. The method proposed below can be described as cross hatching in three dimensions. In addition, our method for cross-hatching is sensitive to lighting conditions. Hence the method belong to the class of non-photorealistic renderers. Other non photorealistic renderers include the work by Saito and Takafumi and Takahashi[11], Haeberli[12], and Hanrahan and Haeberli[13].

Takafumi and Takahashi[11] 'enhance' two dimensional images to produce 'illustrations' of them. The illustrations are non-photorealistic in the sense that the enhancement process ignores the way light reflects off surfaces. Rather it emphasises profiles and edges, and draws cross-hatch contour lines. Haeberli[12] chooses to use a complicated point-spread function when ray tracing. The point-spread function is in the shape of the stroke of a paint-brush, and can be aligned to lay tangent to the surface being rendered. The result is an image that gives the appearance of being painted rather than photographed. This is true also of the images produced by Hanrahan and Haeberli[13], who interactively 'paint' the surface characteristics (colour, unit normal etc.) usually associated with a conventional lighting model onto an object.

The unique feature of the work presented is that it uses a lighting model to adapt a grid, so enabling meaningful cross hatching. The cross hatching operates in three dimensions to texture map the object. Thus it is resembles the work described in both Takafumi and Takahashi[11], and Hanrahan and Haeberli[13], without being either of them.

The claim is that the method can carry both form and colour, in the same image, without interfering with one-another. Results are presented in section 3 to support this claim. The paper concludes in section 4 by suggesting that our structured lighting represents a good solution to the problem in hand and that it could be used in conventional computer graphics as well as visualisation. First, however, the methods must be explained.

2 The structured-lighting model

Our method for simulating structured lighting is very simple; its only requirement is that objects are decomposed into points by the renderer. Given that scan-line renderers and ray tracers both do this, we claim this is not a restrictive requirement. Briefly, the general case model works as follows: use a conventional lighting model to compute the total energy of light that enters the eye. That total energy us used to adapt a 'shadow-grid', so that lower energies lead to denser grids. The lighting model is then 'turned off' and the grid is used is used to texture map the object. Regions of the object that are not touched by the grid do not have their colour effected. The method is now described in greater detail.

To compute the total energy of light entering the eye from a point in the scene being rendered any conventional mechanism may be used. There is no restriction on this. If that conventional model yields a vector (as when computations are wavelength dependent) then the energy over each component should be integrated to give the total.

The regular grid comprises a set of orthogonal planes, spaced at regular intervals; each plane has a width. Adapting the grid as a function of total energy means either (1) effecting the spacing between planes, or (2) effecting the plane width. To compute the interval between planes, gap, given a total energy, E, we use

$$\text{gap} = \text{min\_gap} * 2^{\text{floor}(k*E)}$$

where min_gap is the minimum gap between planes, k is a scale factor that controls how quickly the interval grows with E, and floor(x) is the greatest lower bound of a number, x. This mechanism ensures that any plane drawn in a lighter region will continue into darker regions without breaking or bending. If this computation is performed incorrectly then either of these unwanted effects could happen, as shown in figure 1.

Adapting the plane width, if it is desired, can be performed by an analogous mechanism, except that the width should be reduced in lighter regions. In practice this means using a negative scaling coefficient, k, and a maximum width rather than a minimum gap. Notice that if no surface normal is available then grid cannot be adapted in this way.

Once the grid has been adapted (or not) the point being rendered must be tested against it. So, the next step is to transform the point into ‘grid space’. For three dimensional grids this is performed by a simple matrix map, for two dimensional grids a projection of some kind is necessary. The point can now be tested against the grid.

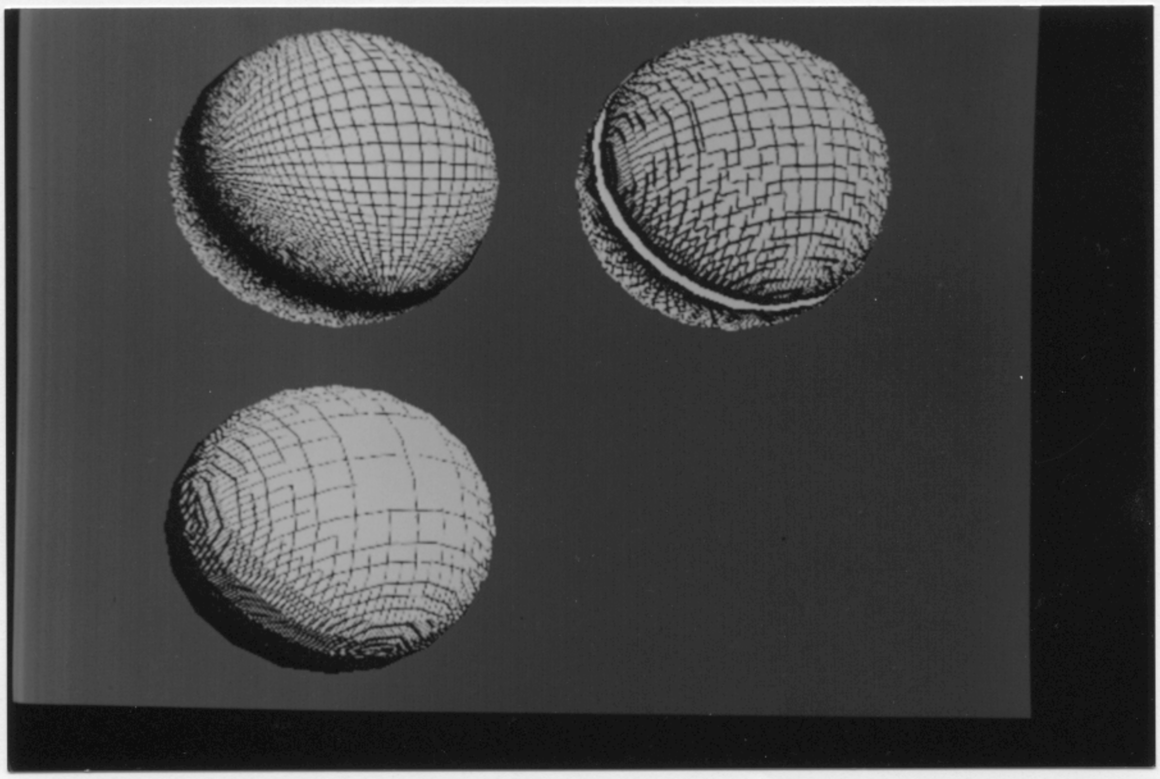

Fig 1 : Effects generated by incorrect adaptation of inter-plane gap.

If the point lies on the grid then the colour of the point is taken to be the colour of the plane; usually this will be black. Otherwise, the colour of the point remains unchanged from its ambient value, say. If the point is closer to any plane in the grid than the half-width of the grid, ω, the point is deemed to lie on the grid. This condition is expressed by

$$\omega \quad (\delta = \min_i(|x_i - \mathrm{floor}(x_i / gap)|)) \text{ implies “point lies on grid”}$$

where x_i is a point coordinate value, and $\min_i$ runs over all components. Once the condition “point lies on grid” has been established a value for the point colour is computed. Typically, this is computed from a linear combination of the grid colour and the original colour of the point. So that

$$C = f(\delta/\omega)C_{point} + (1-f(\delta/\omega))C_{grid}$$

where f(δ,ω) is some function such as δ / ω. Such a linear combination provides an anti-aliasing mechanism that keeps edges of lines looking smooth. The function f(δ/ω) can be any suitable function, such as a Gaussian. The algorithm is summarised, pictorially, in figure 2.

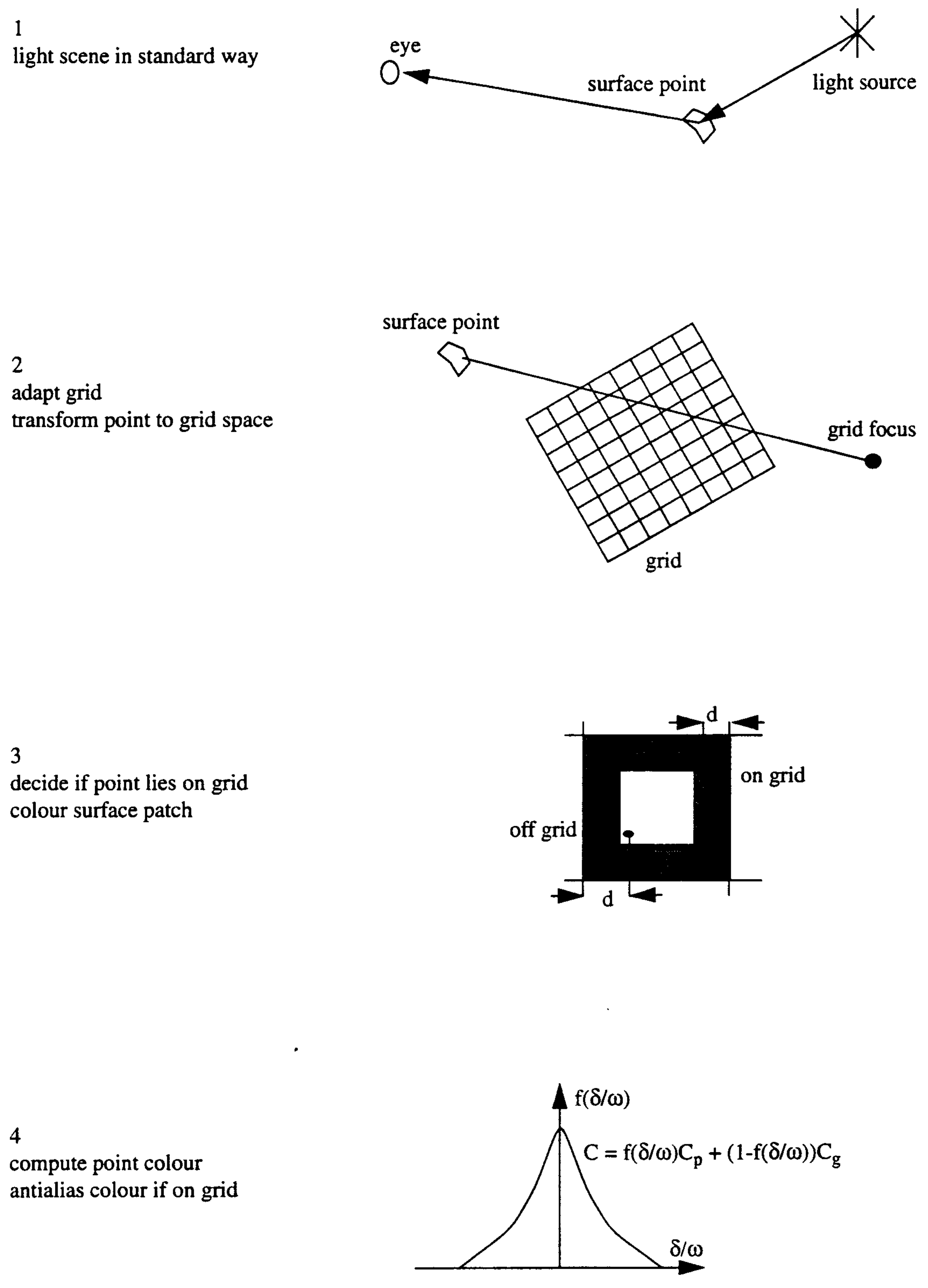
1
light scene in standard way
eye
surface point
light source
2
adapt grid
transform point to grid space
surface point
grid focus
grid
3
decide if point lies on grid
colour surface patch
d
on grid
off grid
d
4
compute point colour
antialias colour if on grid
f(δ/ω)
C = f(δ/ω)C_p + (1-f(δ/ω))C_g
δ/ω

Fig 2 : A visual synopsis of the algorithm

3 Results

The images from figure 3 to figure 5 are all visualisations of a simulated reverse flow pipe combustor, a computational fluid dynamics application. They show the iso-surface that is formed within the combustor by fluid that does not move in a direction parallel to the cylindrical axis of symmetry, and all are coloured according to air speed on the surface (direction of motion must be orthogonal to the axis). The images demonstrate the use of a standard lighting model, a two dimensional adaptive grid, and a three dimensional adaptive grid respectively. Notice how, in some regions of the surface, colour under the standard lighting model is more yellow than under the structured lighting model.

The images in figure 6 and figure 7 show how the three dimensional grid may be used in a volume rendering context. Both of the images are visualisations of the fluid flow in the reverse flow pipe combustor; this is a vector field in three dimensions. The colours in them have been determined to indicate the colour of the vector[8]. There is no surface information available with this kind of rendering, so standard shape cues cannot be used, neither can the grid be adapted. The un-adapted three dimensional grid has been sandwiched between planes of interest in figure 6. The grid plane has been set to black, and the planes demonstrating variable distribution are fully opaque. This arrangement provides a visible 'edge' to the plane and is a shape cue. In figure 7 a three dimensional grid is defined over the entire volume, but appears only where the volume is rendered as fully opaque. Hence is it visible only on the 'surface' of that part of the object. In this case, the silhouette edges of the opaque volume provide shape cues, the grid lines help dis-ambiguate regions interior to these edges.

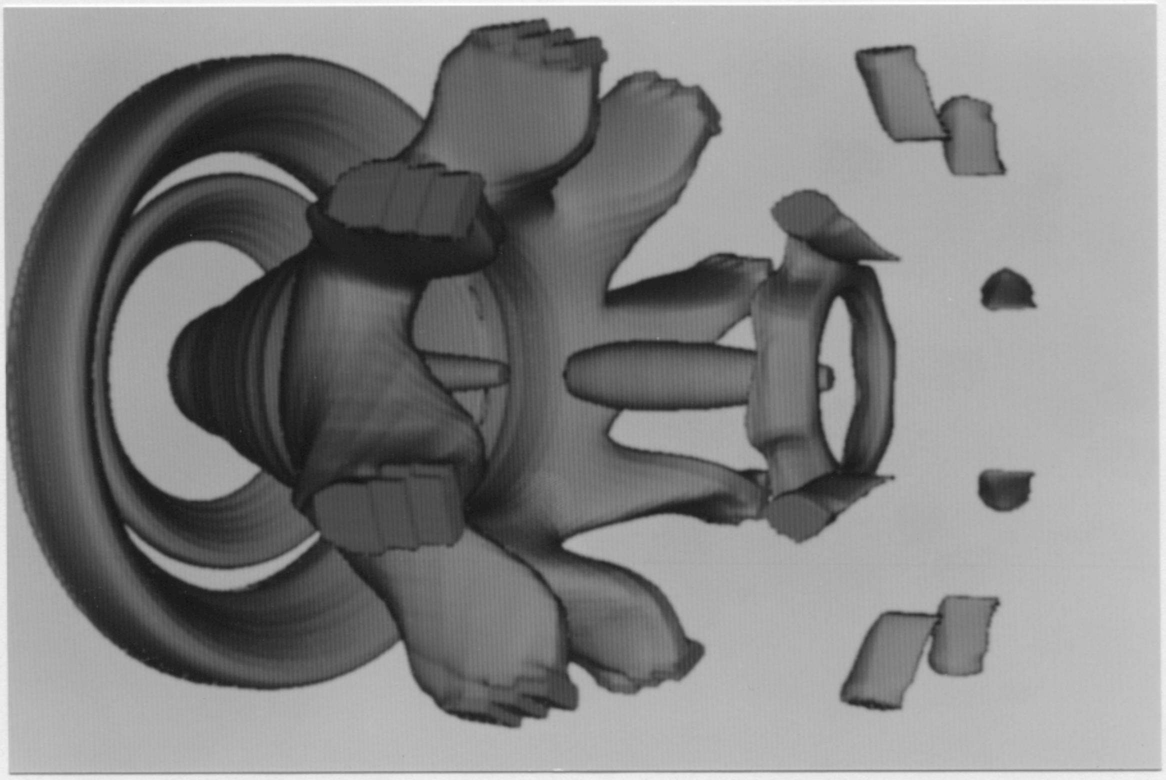

Fig 3 : An iso surface under the standard lighting model.

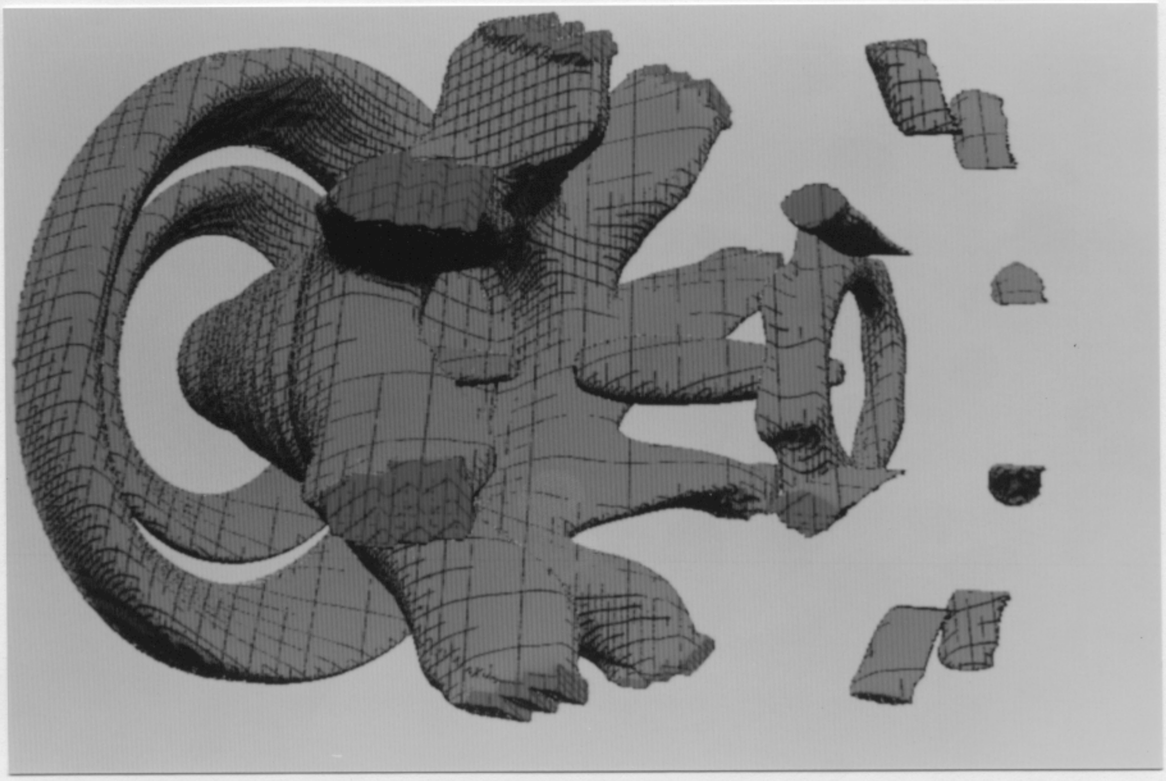

Fig 4 : The iso-surface rendered using a two-dimensional adaptive grid.

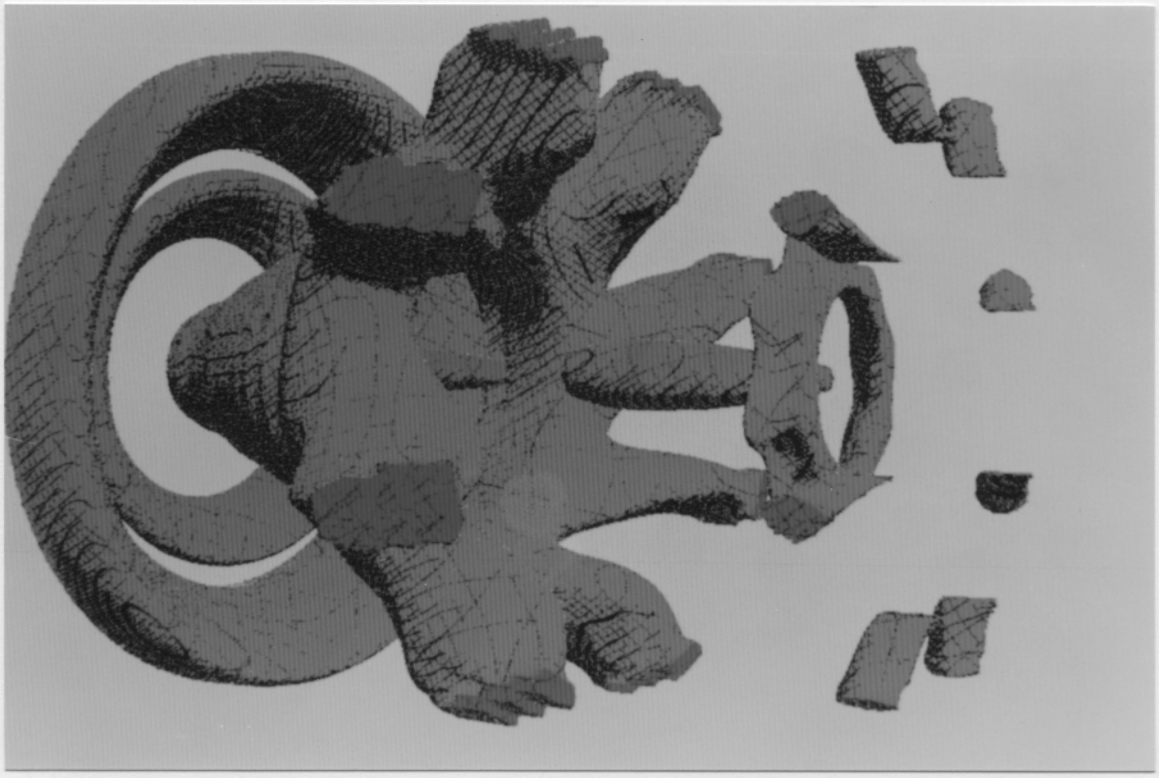

Fig 5 : The iso-surface rendered using a three-dimensional adaptive grid.

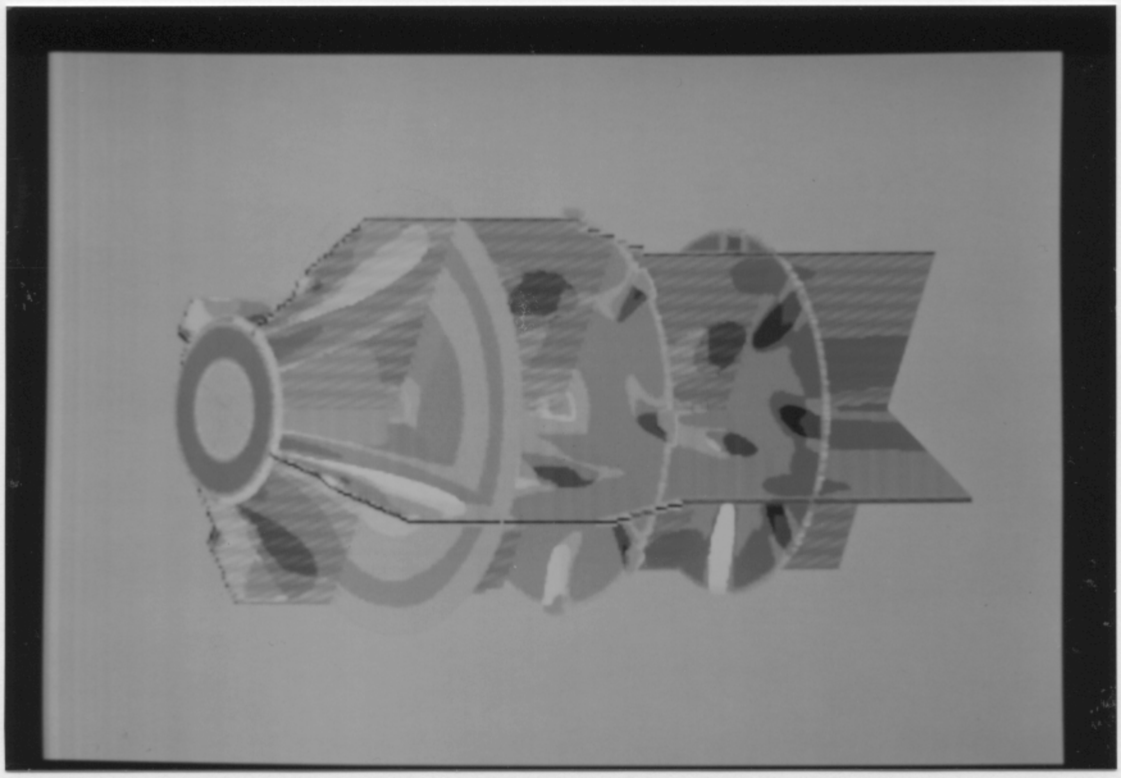

Fig 6 : The fluisd flow rendered as slices, 'sandwich' planes provide shape cues.

Fig 7 : Fluid flow partioned into volumes, a three-dimensional grid provides shape cues.

4 Conclusion

A structured lighting model has been proposed as a suitable replacement for the conventional lighting model in the context of visualisation. The shading model results in images that are shaded in a style reminiscent of cross-hatching. This is true for both two-dimensional and three-dimensional shadow grids. Consequently it belongs to the 'non-photorealistic' class of rendering techniques, and is a distinct member of that class.

The technique trades shape fidelity in order to gain colour fidelity, but in the conventional lighting models this trade-off is reversed. We hypothesise that attribute mapping necessarily involves a compromise between colour and shape fidelity.

The technique has been shown to be applicable to both attribute mapping of surfaces and volume rendering. The contribution made to visualisation is that it separates shape cue from surface colour.

5 Acknowledgments

This work was supported by a SERC quota award.

6 References

[1] Marshall R.E., Kempf J., and Dyer S. (1990) Visualization methods and simulation steering for a 3D turbulence model of lake Erie. *Comput Graph (Proc. SIGGRAPH)* 22(4): 89-97

[2] Wyvill G., McPheeters C., and Wyvill B. (1986) Data structure for soft objects. *The Visual Computer* 2: 227-234

[3] Lorenson W.E., and Cline H.E. Marching cubes: A high resolution 3D surface reconstruction algorithm. *Comput Graph (Proc SIGGRAPH)* 21(4): 163-197

[4] Peachy D.R. (1985) Solid texturing of complex surfaces. *Comput Graph (Proc. SIGGRAPH)* 19(3): 279-286

[5] Perlin K. (1985) An image synthesizer. *Comput Graph (Proc. SIGGRAPH)* 19(3): 287-296

[6] Drebin R.A. Carpenter L. Hanrahan P. (1988) Volume rendering. *Comput Graph 22(4) (Proc. SIGGRAPH)*: 65-74

[7] Levoy, M. (1988) Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, May '88: 29-37

[8] Hall P.M. (1993) Volume rendering for vector fields. *The Visual Computer* 10(2): 69-78

[9] Will P.M., and Pennington K.S. (1972) Grid coding: A novel technique for image processing. *IEEE Proc.* 60(6): 669-680

[10] Hall E.L., Tio J.B.K., McPherson C.A., Draper C.S., and Firooz A.S. (1982) Measuring

curved surfaces for robot vision. *IEEE Computer* December '82: 42-53

[11] Takafumi S., and Takahashi T. (1990) Comprehensible rendering of 3D shapes. *Comput Graph (Proc. SIGGRAPH)* 24(4): 197-206

[12] Haeberli P. (1990) Paint by numbers: abstract image representations. *Comput Graph (Proc. SIGGRAPH)* 24(4): 207-214

[13] Hanrahan P. , and Haeberli P. (1990) Direct WYSIWYG painting and texturing on 3D shapes. *Comput Graph (Proc. SIGGRAPH)* 24(4): 215-224