*Volume xx*, (*200y*), *Number z pp. 000–000*

# Rendering silhouettes with *Virtual Lights*

Domingo Martín and Juan Carlos Torres

Dpto. de Lenguajes y Sistemas Informaáticos
Universidad de Granada
Avda. Andalucía, 38
18008 Granada,Spain
E-mail:[dmartin,jctorres]@ugr.es

**Abstract**

*We present a new method for obtaining non-photorealistic images. These images have two main visual components: silhouettes and non-realistic colouring. Silhouettes are lines that define the form of an object. They are used in classical animation and illustration as the main expressive components. In these applications, if it is necessary, colouring can be added once the drawings are made. For instance, generally, in illustration, colouring is flat and does not transmit volume information whilst silhouettes do it in an economical way. The proposed method is based on the Virtual Lights model, which allows us to use external components, the virtual lights, to define silhouettes. In this way, the designer is free to control where, when and how the silhouettes must appear. The method can be used with B-rep geometric models.*

***Keywords:*** non-photorealistic rendering, silhouettes

## 1. Introduction

Non-photorealistic rendering or expressive visualisation[1] has been an area of great interest over the last few years. Whereas the main goal in photorealistic rendering is the physical accuracy of the image, the objectives of expressive visualisation are the aesthetic appearance and the interpretative capabilities. Expressive visualisation offers advantages with respect to the photorealistic approach for several applications, in which it is more important to represent than to reproduce. Classical animation, illustration and schematic representation are some examples.

The goal of our work is to produce images that look like hand-made ones in an automatic way, but providing the user the freedom to decide which lines will appear for every object in the scene. Firstly, we need to define the elements that we will use to give the hand-made appearance to the image. In general, the main characteristics found in these kinds of images are silhouettes and non-realistic colouring. Silhouettes are lines that define the form of an object (Figure 1). Silhouettes can be used in conjunction with photorealistic rendering. In this case, the colour transmits the information about the shape of an object (Figure 2.c). Silhouettes can be added to this kind of rendering to strengthen the shape information (Figure 2.d). This is not the case for classical animation and illustration in which colour is not used in such a realistic way. Silhouettes transmit the general form of the object, which may be enhanced with the use of a more simplified colouring, normally flat or with only a few shades (Figure 2.b).

Great advances have been made in expressive visualisation in the last few years. A review is given in Lansdown's paper[1]. In general, hidden line elimination methods are related to silhouette rendering, and in fact some of them can be used for this, such as Appel's method[2] or Elber's one[3].

The process of rendering silhouettes can be divided into two stages; a selection stage and an extraction stage. In the selection stage, some parts of the geometric model are *selected* as silhouettes because they satisfy some condition. After that, the selected parts are *extracted* for being drawn.

In the process of selecting the silhouettes, the geometric elements that represent them can be selected manually or automatically. In the former case, the user selects the geometric elements directly depending on what he/she wants to see. In the automatic case, the geometric components are selected depending on a condition that may relate external components of the scene (observer, light, etc) with the object (ge-

**Figure 1:** *Example of silhouettes (outlines and shape lines).*

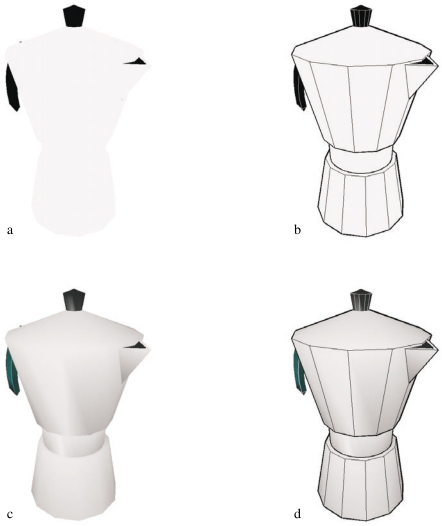

**Figure 2:** *The effect of silhouettes over a flat coloured model (b), and a Gouraud shaded model (d).*

ometry, material, etc), or that may depend only on the object (geometry, parametric values, material, etc). The first type of condition is generally used to select the outlines, using the position of the observer. Other silhouettes, which depend on the colour, are selected using lights. In this way, the observer and the lights determine not only the appearance of the silhouettes, but also what the observer sees, the object colour and other effects. That is, the silhouettes are the result of illuminating the scene, and locating the observer to see it.

The main advantage of the method that we present is that it separates the location of the observer and the lighting process from the selection of silhouettes. This is done using external components, the virtual lights, which specify the silhouette's location in a more flexible way. The basic idea is to use virtual illumination, which produces virtual changes in shadows and colour; these changes are then used to define the silhouettes.

The method unifies the selection of different types of silhouettes, enabling the user to define and control where, when and how they will appear, allowing the use of several 3D geometric models. As we are especially interested in producing images with a 2D appearance (classical animation and illustration), all the examples are oriented towards this kind of application.

A basic version of the *Virtual Lights* model was presented in[25]. The model presented had limitations because it was defined only for polygonal models, and the use of it was highly restrictive.

## 2. Related works

Most of the related works commented below use the "normal" approach of selecting the silhouettes with the location of the observer and the lights.

We classify the methods, depending on the space in which the silhouettes are obtained, as 3D methods, 2D raster methods, and mixed ones[4]. In 3D methods, selection and extraction of silhouettes is achieved in 3D space. Raster methods use the information saved in a raster-like data structure. If the information saved does not refer to 3D information, the method is 2D, otherwise it is a 3D+2D method. There is a trade-off between flexibility and speed. Whereas a 3D method has more possibilities, though the process of extracting is normally slow, a 2D method is faster (even hardware accelerated) but less flexible.

Salisbury et al. present a 3D method[5]. They use a standard illumination model to produce the shades used to select the different textures and strokes, together with a planar map to maintain adjacency information, which is also used to determine some characteristics of the strokes. Other related works are[6, 7, 8, 9]. The method proposed by Markosian et al. can be classified as 3D[10]. They implement an enhanced version of Appel's method[2], with the main goal of producing silhouettes in real-time. This is also the objective of Elber's paper, though he uses free form polynomial and rational surfaces[11]. Toledo applies textures in cell animation, using the detection of outlines[12]. Sousa's papers are mainly oriented towards the appearance of the silhouettes, simulating a graphite pen[13, 14]. Northrup allows a great variety of styles of silhouettes[15]. Buchanan presents a data structure, the edge buffer, which

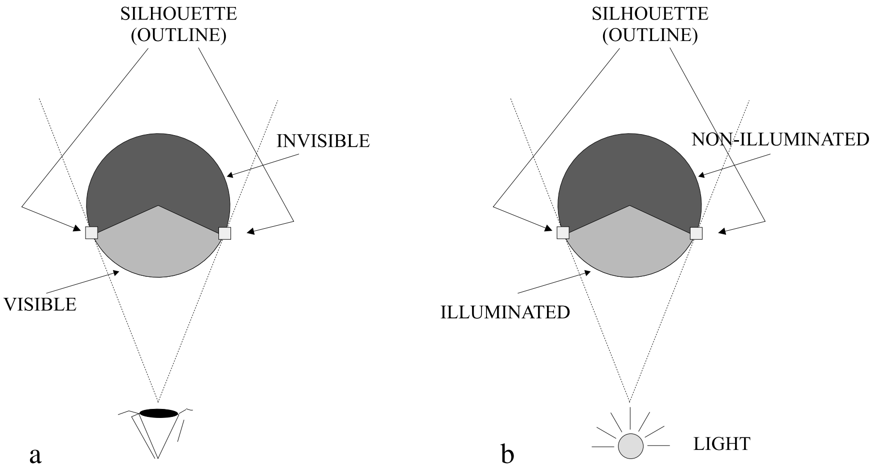

**Figure 3:** *Outlines can be seen as the limit between illuminated and non-illuminated parts of an object.*

is based on the orientation of faces to compute the values[16]. Another approach is to attach the silhouettes to a simple geometry, as used with *graftals*[17, 18] and their extension, the *geograftals*[19].

In the 3D+2D category, the work of Saito and Takahashi has inspired many others[20]. They use the G-buffers to save 3D information that is post-processed to obtain the silhouettes. In fact, the method uses border detection, based on Sobel's operator, applied to the z-buffer. This is a method in which geometric-only conditions can be used (e.g. parametric values of the surface). The Piranesi system deals with the same type of lines[21], adding a material buffer that allows other characteristics to be applied.

A 2D method is presented in[22] that works with the stencil and frame buffers to obtain the outlines. Akeley's method allows them to obtain silhouettes[23].

In relation with the appearance of silhouettes, Hsu's work is a 2D system, but the method could be used once the silhouettes are projected[24].

## 3. The *Virtual Lights* model

We provide a general description of the method before presenting the formal definition for the *Virtual Lights* model.

An economical way to represent objects is to use lines that show their forms, the silhouettes. We select two kinds of silhouettes: outlines and shape lines (Figure 1). Given an observer, the outlines are the lines that represent the limit between the visible and invisible parts of the object. Given one or more lights, shape lines are lines that represent the limit between different colours or shades.

The basic idea of the *Virtual Lights* model is that, just as light can be used to produce changes in the colour and/or shadows of objects, something similar can be applied to selecting silhouettes. In this way, the user could define and control, in an easy and flexible way, where, when and how the silhouettes will appear, using an external component similar to lights.

Outline and shape line silhouettes have one characteristic in common: they can be expressed as limits between visual properties. We are interested in including both types of silhouettes in the same formalism using this characteristic. This can be done if outlines can also be controlled using lights. The key for integrating outlines is to assume that they are limits, but not between visible and invisible parts, in the sense of an observer (Figure 3.a), but between illuminated and non-illuminated parts, for a light located at the same position as the observer (Figure 3.b). It is important to note that being visible and being illuminated from a light's point of view is the same thing.

The importance of this approach is that we can use lights to define and control the outlines and the shape lines of an object. But, if we use real lights to define the silhouettes, they will also change the colour of the object, and we are only interested in producing the silhouettes. So, we need to develop new components that are used to select only the silhouettes: the *virtual lights*. The difference between real lights and virtual lights is that virtual lights produce changes that are used to obtain the silhouettes whilst real lights produce changes in colour and shadows. As virtual lights do not produce changes in the real colour of an object, we can define as many virtual lights as we wish to produce the silhouettes necessary for the desired result. In fact, every object has its own set of virtual lights.

As in the case of real lights, the virtual lights are used to compute the reflected intensity on which the silhouettes depend. We use a reflection model based on the simple il-

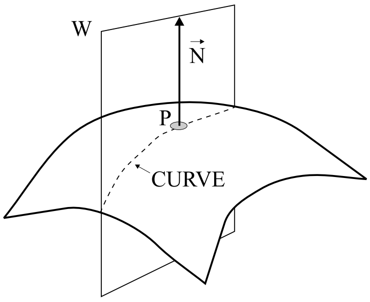

**Figure 4:** *The curvature is computed using the curve obtained as the intersection between the plane W and the surface.*

lumination model[2] in which the reflected intensity of an object depends on three components, $I = I_{ambient} + I_{diffuse} + I_{specular}$.

In the *Virtual Lights* model, the virtual lights can be classified depending on the kind of reflection used to compute the silhouettes, that is, there are diffuse and specular virtual lights, each one with its associated reflection model. We have not included the ambient virtual light because there is not an ambient reflection model that can be used to obtain the silhouettes.

The virtual lights can also be classified depending on the effects they produce, related to those of the components of the simple illumination model. Thus, a diffuse virtual light produces silhouettes that depend on the position of the virtual light, but not on that of the observer. A specular virtual light produces silhouettes that depend on the position of the virtual light and of the observer.

We can now give a formal definition of the *Virtual Lights* model, which is made up of several elements:

- A valid 3D geometric model
- Virtual lights
- Illumination models

The concept of curvature must be explained before introducing other concepts. We use the concept of normal curvature[26]. Given a point $P$ on a surface, the normal vector $\vec{N}$, and a plane $W$ that includes the normal vector (Figure 4), we can rotate the plane $W$ around the normal producing curves that are the intersection between the plane and the surface. Given an orientation for the plane $W$, the curvature can be computed for the curve obtained at point $P$. This curvature is used as a geometrical condition, defined as an interval $C(c_1, c_2)$. The curvature condition is satisfied when the computed curvature lies within the values of the interval. If $c_1$ and $c_2$ are equal, we call this condition *fixed curvature*, otherwise, we call it *relaxed curvature*.

**Definition 1** A 3D geometric model is valid when it supports the possibility of computing the normal vector and the curvature for every point on the object surface.

**Definition 2** A virtual light $VLP$ is a point in homogeneous coordinates $(x, y, z, w)$.

If $w = 0$, the virtual light is located at infinity, and called *non-local* virtual light. If $w \neq 0$, the light is near the scene, and called *local* virtual light.

**Definition 3** A diffuse illumination $DIM$ model computes a real value I between -1 and 1 for a virtual light and for every point of the object's surface, with the expression:

$$I = cos\theta$$

Where $\theta$ is the angle between $\vec{N}$, the normal vector of the surface, and $\vec{L}$, the direction vector to the virtual light. A diffuse illumination model is associated to a virtual light to produce silhouettes that are independent of the observer's position.

**Definition 4** A specular illumination model $SIM$ computes a

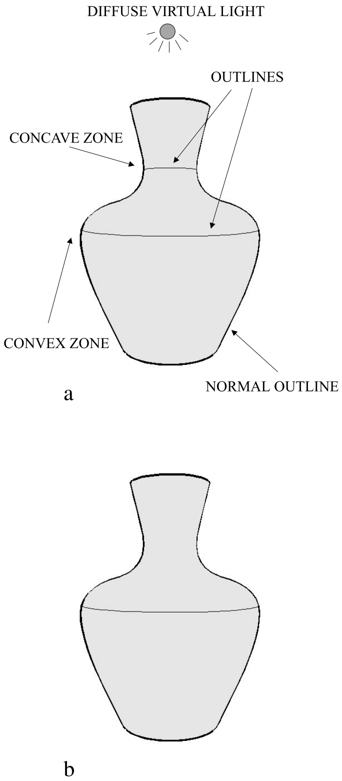

**Figure 5:** *We can select which lines will be seen depending on the curvature. All the outlines are visible in (a), whilst those in a concave zone, for the virtual light on top, are removed in (b), based in aesthetic choice of the user.*

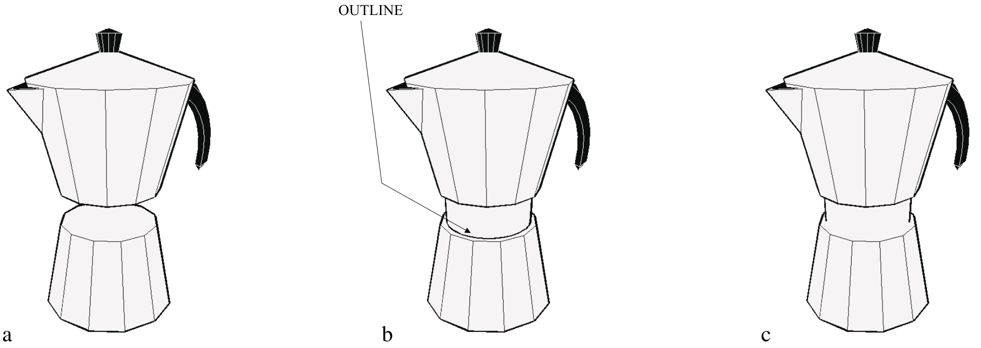

**Figure 6:** *We can control whether some edges can be selected as silhouettes (b) or cannot be selected (c).*

real value I between -1 and 1 for a virtual light and for every point of the object's surface, with the expression:

$$I = cos\alpha$$

Where $\alpha$ is the angle between $\vec{R}$, the symmetric vector to $\vec{L}$ in relation to $\vec{N}$, and $\vec{V}$, the observer direction vector. A specular illumination model is associated to a virtual light to produce silhouettes that are independent of the virtual light and the observer's position. One difference from the specular component of the simple illumination model is that the cosine function does not have an exponent that represents the brightness of the object, because, as shown below, the same effect can be produced in a different way.

Because we are interested only in relative changes of visibility or reflection, the intensity of the lights and the reflection coefficients can be simplified.

As previously explained, the curvature depends on a curve which itself depends on the orientation of the $W$ plane. We compute the curvature using the curve obtained when the normal of plane $W$ is perpendicular to $\vec{L}$.

**Definition 5** Given a 3D geometric model, a virtual light, a diffuse illumination model *DIM*, and a curvature condition $C$, an outline is the set of points that satisfies that the computed reflection of each point is equal to 0 and the curvature satisfies the condition $C$.

That is to say, outlines of an object are the limits between the illuminated and non-illuminated parts of the object, limiting these parts with the curvature condition.

The curvature condition is used to select the silhouettes from concave zones, convex zones, or from both types of zones, using, usually, a relaxed curvature. Generally, this is an aesthetic option, that is, the user will select the silhouettes of concave and/or convex zones, depending on personal taste. For example, in the Figure 5, we want to make clear that the object has a volume because it is drawn with a flat colour model which does not indicate the form of the objet. This concave object has defined two diffuse virtual lights: one at the observer's position and another above. The virtual light above produces outlines in the convex and concave zones of the object that are visible to the observer. In this example, we have decided only to allow the outlines of the convex zone, based in our aesthetic taste. The "normal" outlines form a special case: these are produced when a diffuse virtual light is located at the observer's position. The special characteristic is that normal outlines of concave zones are invisible, because they are hidden by other parts of the object. Each object must have only one diffuse virtual light to produce the normal outlines, and it must be the same for all the objects.

**Definition 6** Given a 3D geometric model, a virtual light, an illumination model, *DIM* or *SIM*, a curvature condition $C$, and one real number $R \neq 0$, a shape line is the set of points that satisfies that the reflection is equal to $R$ and the curvature satisfies the condition $C$.

It is now clear that the effect of the exponent of the specular component of the simple illumination model can be achieved by the real value $R$ with virtual lights.

In some cases, it may be necessary to define silhouettes that are dependent on geometric conditions or independent of any condition. The geometric conditions can produce silhouettes that represent special characteristics in the form of the object, for example, a highly convex or concave zone, wrinkles, curves with the same parametric value, etc.

In other cases, the user can control the appearance of silhouettes directly. The user marks which parts of the geometry will appear as silhouettes, regardless of the conditions. This capability also includes the possibility of marking parts of the geometry that cannot be selected as silhouettes. This option is especially useful when there is a combination of pieces. Silhouettes that are not necessary can appear in the zone where the pieces are joined. In this case, the contact zone between the two pieces is marked as forbidden for the selection of silhouettes (Figure 6).

As previously commented, each object has a different

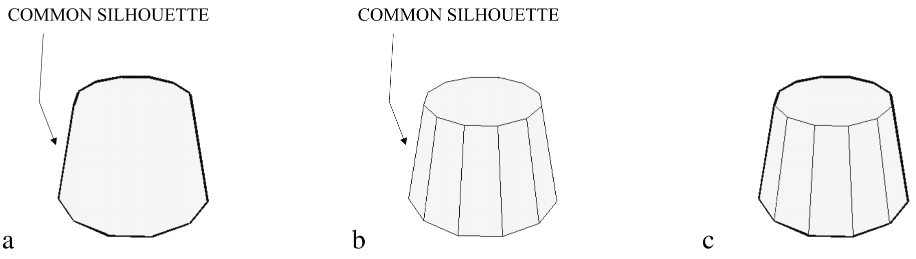

**Figure 7:** *Defining a priority when a shape line is selected several times. There is a collision between the silhouettes of (a) and (b). The silhouettes in (a) have priority over those in (b). The final result is shown in (c).*

set of virtual lights, as well as geometric conditions, and each virtual light and geometric condition can have different drawing attributes. As several virtual lights and geometric conditions may select the same set of points, it is necessary to decide which attributes will be used. Establishing a priority, by which every set of points has an identifier of the virtual light that produces it, can do this. Once a priority has been associated to a set of points, the same one cannot be rendered with another. In Figure 7, the object has one virtual light and one geometric condition which mean that some silhouettes are selected twice, once for the diffuse virtual light located at the observer's position (a), and again for the geometric condition (b). The user determines that the outline has priority over other silhouettes (c).

## 4. The implementation

The general model for virtual lights has been developed using a 3D geometric model based on flat polygons, specifically triangles, using the Winged-Edged data structure[27] (the use of triangles is due to the need of computing the normal vector, but other types of polygons could be used, if it is guaranteed that the model is valid).

In general, polygonal models represent objects as an approximation. Their main advantages are their simplicity and the existence of techniques developed for flat polygons, especially triangles, as well as special hardware to accelerate the rendering process and libraries that use triangles as basic graphic primitives, for example, OpenGL. Moreover, computing the normal vector and the curvature are easily achieved.

The triangle-based polygonal model allows easy classification of the geometry that forms the silhouettes. In this case, edges are the components that form the silhouettes (the introduction of new edges is not allowed). The problem is that whilst it is possible to find a set of points that match a condition, defined by a single value, when a continuous surface is used, we cannot do the same with a surface approximated by a polygonal model. For example, given a single value for a normal vector, there may be no faces with that value. Thus, for polygonal models, the single values have to be converted to intervals, $(value_1, value_2)$.

We need to adapt the definitions of the *Virtual Lights* model to the special conditions of polygonal representation. All the computations are applied to faces and the silhouettes are formed by one or more edges. Given an edge and its two faces, the reflection value of the edge is obtained by computing the reflection difference between the faces. The curvature at the edge is the curvature between the faces, which is computed as the angle between the two normal vectors.

We need to define when a face is illuminated or non-illuminated, prior to the re-definition of silhouettes (as previously commented the terms visible and illuminated are interchangeable for a virtual light).

**Definition 7** A face is illuminated for a virtual light if the dot product between the normal vector of the face and the direction to the virtual light is greater than 0. Otherwise, the face is non-illuminated.

**Definition 8** Given a 2-manifold 3D polygonal model, a virtual light, a diffuse illumination model *DIM*, and a curvature condition $C(c_1, c_2)$, an edge is marked as an outline if it has one face illuminated, the other face non-illuminated, and the angle between the two faces satisfies condition $C$.

The curvature is implemented as an angle interval. The angle interval limits the value of the angle between the normal vectors of the two faces. The concave condition is expressed as an angle interval between 0 and 180 degrees (Figure 8). The convex condition is expressed as an angle interval between 180 and 360 degrees. If the angle is between 0 and 360 degrees, the curvature has no influence.

**Definition 9** Given a 2-manifold 3D polygonal model, a virtual light, an illumination model, *DIM* or *SIM*, a curvature condition $C(c_1, c_2)$, and an interval of real numbers $R(r_1, r_2)$, an edge is marked as a shape line if the angle

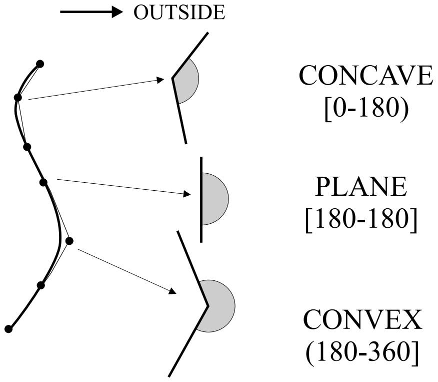

**Figure 8:** *The curvature is implemented as an angle interval.*

between faces satisfies condition *C* and the reflection is included in interval *R*.

As we stated before, sometimes is better to select the silhouettes using geometric conditions. We have implemented only geometric conditions that use the angle interval and the principal curvature, which is the computed angle between the normal vectors of the two faces. For example, a geometric condition allows us to select all the edges that give its typical appearance to the base of the coffee maker. It would require several diffuse virtual lights to do the same.

Each edge has an attribute that marks whether it can be selected or not as a silhouette, indicating whether the edge is always a silhouette, never is or depends on the operations with virtual lights. This attribute can be used for direct selection of edges or to prohibit some of them.

The use of a 3D polygonal model can produce problems with the continuity of silhouettes, namely a temporal-continuity problem and a space-continuity problem. The temporal-continuity problem is related to the sudden appearance of silhouettes when the observer or virtual lights are in movement. This is not a problem only for silhouettes and virtual lights: it is inherent to the model itself, depending on the level of approximation, although this is not a problem when the object represented is polygonal. The space-continuity problem is related to the staircasing appearance that silhouettes can have when the polygonal model is only an approximation of a curved object. The same problem occurs when illumination is applied to an object and it appears to be faceted.

### 4.1. Extracting the silhouettes

Once the edges are selected, using the *Virtual Lights* method, they have to be extracted and drawn as silhouettes. This process is a secondary goal for us because there are very fast methods for doing that (e.g. Markosian's one[10]). Instead, we are more interested in a simple method that is flexible enough. We have implemented two methods.

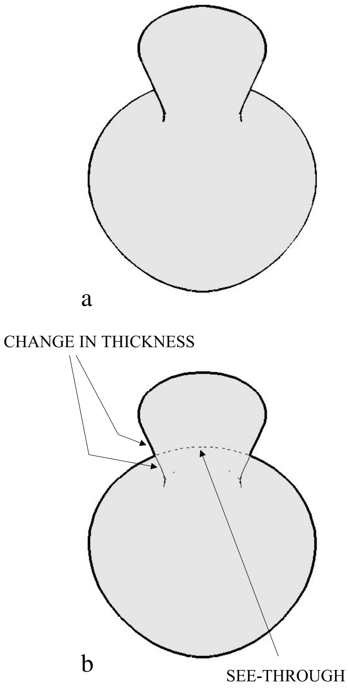

**Figure 9:** *The effect of the two rendering methods. All the silhouettes have the same thickness (a). Outer silhouettes are wider than inner ones and hidden parts are also visible (b).*

The basic method traverses the list of edges taking only those which are selected. The edges are classified depending on the virtual light that produces them, taking into account the priority given. The selected edges for every virtual light are then joined to form chains, which can usually be considered as the representation of a complete shape line, for example, the normal outline of a convex object. The chains are formed with edges that have a common vertex, with extensive use of the winged-edged data structure. The chains can be open or closed. There are cases in which there are several candidates to be taken, especially with open objects or when the object is poorly approximated. We have implemented two possibilities: to obtain outermost chain or the innermost one. In both cases, a hidden part elimination method must be applied. In our implementation, based on OpenGL, the z-buffer is activated. Though very fast and easy to implement, there are problems with wide lines. This problem appears even when the *glPolygonOffset* is active, because the solution is dependent on the orientation of the face: the more perpendicular to the observer the face is oriented, the better the wide line is rendered. This problem can be resolved with Wang's method[28].

The second method is more powerful because it computes the intersections between silhouettes, which adds a higher level of flexibility when rendering them. In animation, and especially in illustration, the appearance of silhouettes depends on the relative position: if they are outside lines they are drawing wider than inside lines. In Figure 9 there is an example showing this characteristic: all the silhouettes have the same width (A), or they change (B). The current implementation has quadratic complexity, related to the number of faces. This method can be improved to a complexity of $\mathcal{O}((n+k)\log n)$, where n is the number of segments and k is the number of intersections[29]. The edges are joined forming chains in the same way as the first method. The most important advantage of this method is that it has no problems with wide and very wide lines, because all the silhouettes are in a plane in front of all the elements of the scene.

## 5. Using *Virtual Lights*

We now show the capabilities of virtual lights with some examples. Firstly, we use a coffee maker that is composed of several pieces (Figure 11.b) each of which has its own set of virtual lights. The virtual lights are added to produce different silhouettes. The positions of all the virtual lights are shown in Figure 11.f.

Figure 11.a, shows the flat coloured coffee maker. First we add a diffuse virtual light located at the same position as the observer (position 1) to produce the normal outlines. The result is shown in Figure 11.b. Then, a geometric condition is defined for the base component of the model (piece number 1) which requires an edge to be selected when the curvature is included in the interval of $(200, 360)$ degrees, which represents convex zones (Figure 11.c). In Figure 11.d, two diffuse virtual lights have been defined for the two top components of the model (pieces 3 and 4). The two virtual lights are located above the pieces (position 2). They have been defined for producing outlines. One important characteristic is that every virtual light defined has a colour and a thickness associated (see the handle in Figure 1, and the table or the plates in Figure 17). Another two diffuse virtual lights have been defined for Figure 11.e, which produces shape lines for the same pieces, located at position 3. The shape lines are obtained because the difference of reflection between the two faces of an edge is included in the $(0.4, 1)$ interval.

Finally, we add a specular virtual light for piece 3, the effect of which is dependent on the observer's position. We thus need to produce an animation sequence, changing the position of the observer, from which we have extracted some images to see the effect, as shown in Figure 12. The specular virtual light is located at position 4. The goal of this example is to show a technical capability, not to reproduce aesthetic effect.

The same process is applied to a curved object, a teapot, which is also composed of several pieces. In Figure 13.b, normal outlines are produced with diffuse virtual lights located at the observer's position (position 1). Two geometric conditions are then applied to the handle and the spout (Figure 13.c). In Figure 13.d, two diffuse virtual lights, located at positions 2 and 3, produce outlines. In Figure 13.e, another diffuse virtual light is added to the body of the teapot to produce outlines.

The effect of adding a specular virtual light is shown in Figure 14. The specular virtual light is located at position 5. We have produced an animation from which we have extracted some images, so it is difficult to see the result compared with the animation. We can see that the shape lines appear in Figure 14.b. In the next two images, the position of the teapot changes whilst the outlines remain fixed, as the virtual lights are not moved.

Figure 15 shows an apple, with shape lines representing a bright zone in two ways. In one case, the shape lines limit the highlight, and in the second case, the highlight is filled with shape lines. When the resolution is high, the second possibility allows us to use silhouettes as hatching.

Figure 16 shows a water pot, or "pipo", rendered in different ways. Finally, Figure 17 shows a scene with several objects, each one rendered in a different way. The legs of the table, as well as the board, are obtained using a diffuse virtual light and a geometric condition; one for the silhouette and the other for the inside borders, which are drawn thinner. The plate on the right has one diffuse and two specular virtual lights. The first one is used for the silhouettes, while the second ones add the effect of the shape lines on the concave an convex zones (they change as the observer moves). The plate on the left has four virtual lights; one diffuse virtual light is for the outline, another one is for the white shape line of the interior, one specular virtual light is for the black shape line of the interior, which is wider, and another one is for the white shape lines near the black border. The characteristics more important of the "pipo" is that one diffuse virtual light is used for producing the "shadow" on the right, and a specular virtual light is used for obtaining the effect of a highlight. Finally, the apple is rendered in a realistic way, with shading, but adding the non-realistic components of the outline, using a diffuse virtual light, and the highlight using a specular virtual light.

We have implemented a rendering method that attempts to make the task of locating the virtual lights more intuitive. In fact, locating virtual lights to produce silhouettes is similar to locating real lights to produce changes in colour and shadows. The method achieves its goals by rendering the faces that will form the silhouettes in bright colours (Figure 10).

## 6. Conclusions

We have presented a method to produce images with an expressive appearance, which in classical animation and illustration is generally obtained by using silhouettes and non-

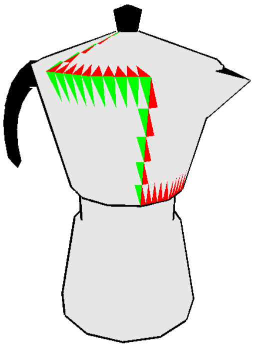

**Figure 10:** *Example showing the selected faces for a diffuse virtual light.*

realistic colouring. The silhouettes can be used to improve a real image, to present a scheme, or to produce 2D animation and illustration.

The model presented allows us to treat outlines and shape lines in a unified way. It uses the same mechanism as the lighting process: the computed reflection. The main contribution consists of the division between the elements of the condition that controls the selection of silhouettes, and the condition itself. Whilst most current methods use the position of the observer to select normal outlines, and lights for the shape lines, this is only one of several possibilities using the *Virtual Lights* model. We can produce the same results as other methods, but we can also obtain new ones. This can be done easily because the process is very similar to locating lights in order to produce light and shade. All the methods that use lighting to select the silhouettes can use virtual lights (though not methods that use any other characteristics unrelated to lighting, such as the type of material).

The use of this method can be improved by a graphical user interface that allows the user to see, at interactive rates, how changing the position and parameters of virtual lights affects the appearance of silhouettes.

The problem of continuity in silhouettes can be resolved as in the illumination case, that is, using an interpolation method, similar to the Phong one.

## 7. Acknowledgements

The authors thank the referees for their useful comments and suggestions that have contributed to improve this paper. This work has been funded in part with grant number TIC98-0973-C03-01 from the Spanish government.

## References

1. J. Lansdown and S. Schofield, “Expressive rendering: A review of non-photorealistic techniques”, *IEEE Computer Graphics and Applications*, pp. 29–37 (1995).
2. J. Foley, A. van Dam, S. Feiner, and J. F. Hughes, *Computer Graphics: Principles And Practice, 2 Edition*. Addison-Wesley, (1992).
3. G. Elber and E. Cohen, “Hidden curve removal from free form surfaces”, *ACM Computer Graphics*, **24**(4), pp. 95–104 (1990).
4. D. Martín, Alhambra*, un modelo para la producción de animación bidimensional*. PhD thesis, Escuela Técnica Superior de Ingeniería Informática, (1999).
5. M. P. Salisbury, S. E. Anderson, R. Barze, and D. H. Salesin, “Interactive pen and ink illustration”, *Proceedings of SIGGRAPH*, pp. 101–108 (1994).
6. G. Winkenbach and D. H. Salesin, “Computer generated pen and ink illustration”, *Proceedings of SIGGRAPH*, pp. 469–476 (1994).
7. M. P. Salisbury, C. Anderson, D. Lischinski, and D. H. Salesin, “Scale-dependent reproduction of pen-and-ink illustration”, *Proceedings of SIGGRAPH*, pp. 461–468 (1996).
8. G. Winkenbach and D. H. Salesin, “Rendering parametric surfaces in pen and ink”, *Proceedings of SIGGRAPH*, pp. 469–476 (1996).
9. D. Martín and J. C. Torres, “*Alhambra*: A system for producing 2D animation”, *Computer Animation 99*, pp. 38–47 (1999).
10. L. Markosian *et al.*, “Real-time non-photorealistic rendering”, *Proceedings of SIGGRAPH*, pp. 415–419 (1997).
11. G. Elber, “Interactive line art rendering of freeform surfaces”, *Computer Graphics Forum*, **18**(3), pp. C1–C12 (1999).
12. W. Toledo, R. Jensen, C. Thayer, and A. Finkelstein, “Texture mapping for cell animation”, *Proceedings of SIGGRAPH*, pp. 435–446 (1998).
13. M. C. Sousa and J. W. Buchanan, “Computer-generated graphite pencil rendering of 3D polygonal models”, *Computer Graphics Forum*, **18**(3), pp. 195–207 (1999).
14. M. C. Sousa and J. W. Buchanan, “Observational models of graphite pencil materials”, *Computer Graphics Forum*, **19**(1), pp. 27–49 (2000).
15. J. Northrup and L. Markosian, “Artistic silhouettes: a hybrid approach”, *Proceedings of NPAR 2000*, pp. 31–37 (2000).
16. J. W. Buchanan and M. C. Sousa, “The edge buffer: a data structure for easy silhouette rendering”, *Proceedings of NPAR*, pp. 39–42 (2000).

17. M. A. Kowalski *et al.*, “Art-based rendering of fur, grass, and trees”, *Proceedings of SIGGRAPH*, pp. 433–438 (1999).
18. L. Markosian *et al.*, “Art-based rendering with continous levels of detail”, *Proceedings of NPAR 2000*, pp. 59–66 (2000).
19. M. Kaplan, B. Gooch, and E. Cohen, “Interactive artistic rendering”, *Proceedings of NPAR*, pp. 67–74 (2000).
20. T. Saito and T. Takahashi, “Comprehensible rendering of 3D shapes”, *ACM Computer Graphics*, **24**(4), pp. 197–206 (1990).
21. S. Schofield, “Piranesi: A 3-D paint system”, *Eurographics U. K.*, (1996).
22. P. Rustagi, “Silhouette line display from shaded models”, *Iris Universe*, pp. 42–44 (1989).
23. K. Akeley, “Algorithm for drawing boundary plus silhouette edges for a solid.” Personal communication, (1998).
24. S. C. Hsu and I. H. Lee, “Drawing and animation using skeletal strokes”, *Proceedings of SIGGRAPH*, pp. 109–118 (1994).
25. D. Martín and J. C. Torres, “*Virtual Lights*: A method for expressive visualisation”, *EG99-SP*, pp. 67–70 (1999).
26. M. E. Mortenson, *Geometric Modeling*. John Wiley & Sons, (1985).
27. B. G. Baumgart, “A polyhedron representation for computer vision”, *Proceedings of National Computer Conference*, pp. 589–596 (1975).
28. W. Wang, Y. Chen, and E. Wu, “A new method for polygon edging on shaded surfaces”, *Journal of Graphics Tools*, **4**(1), pp. 1–10 (1999).
29. F. P. Preparata and M. I. Shamos, *Computational geometry: an introduction*. Springer Verlag, (1985).

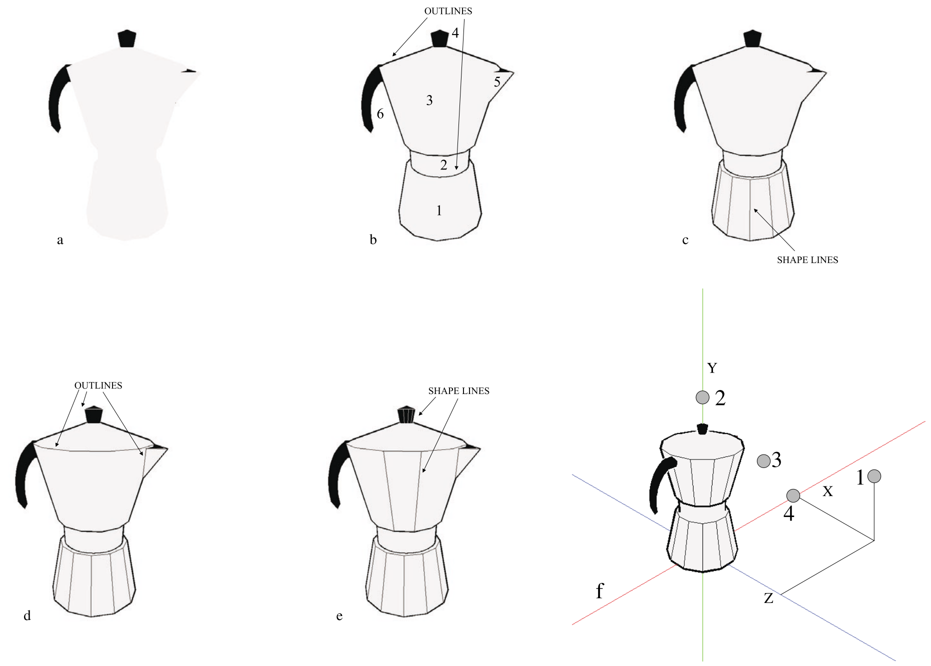

**Figure 11:** *Application of a geometric condition and diffuse virtual lights to the coffee maker. (a) Flat coloured. (b) Selecting outlines using diffuse virtual lights at position 1. (c) Selecting silhouettes using a geometric condition. (d) Selecting outlines using diffuse virtual lights at position 2. (e) Selecting shape lines using diffuse virtual lights at position 3. (f) Position of the virtual lights.*

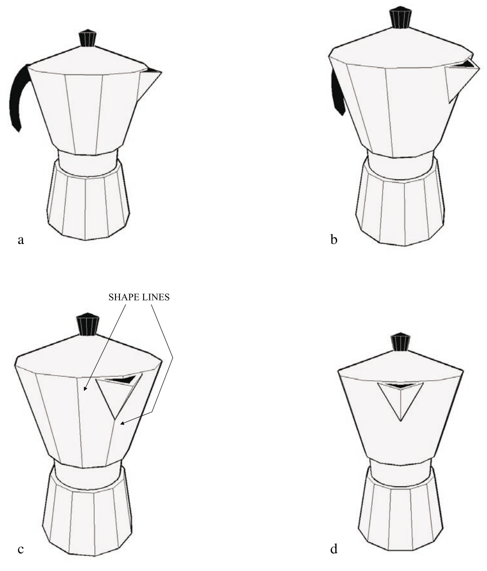

**Figure 12:** *Application of a specular virtual light to the coffee maker. Selecting shape lines using a specular virtual light at position 4. In c, the shape lines appear. (The example show a capability, not necessarily an easthetic effect).*

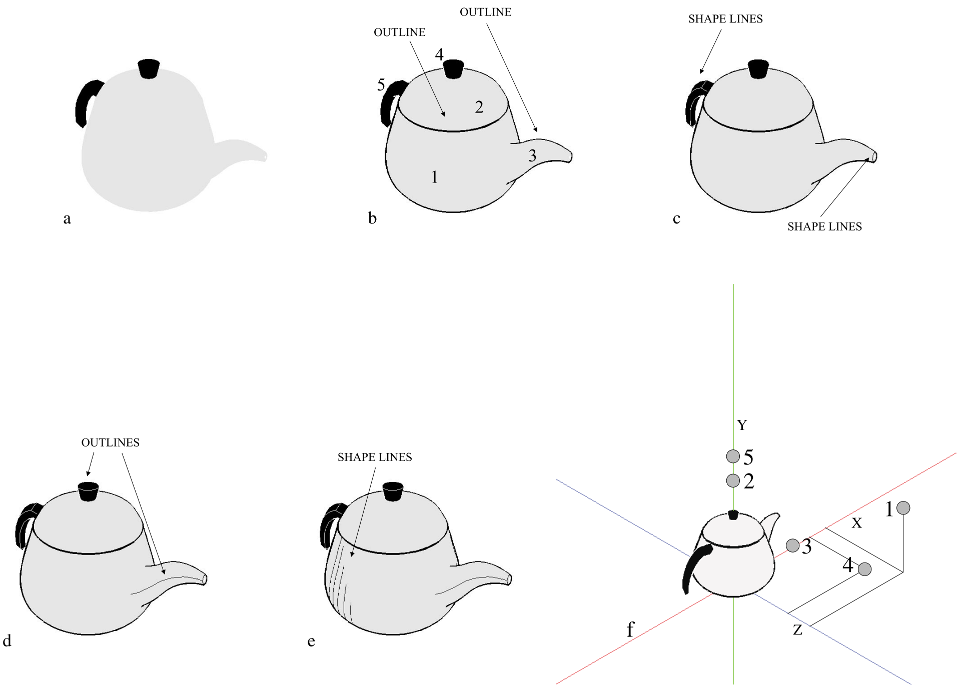

**Figure 13:** *Application of a geometric condition and diffuse virtual lights to the teapot. (a) Flat coloured. (b) Selecting outlines using diffuse virtual lights at position 1. (c) Selecting silhouettes of the handle and spout using a geometric condition. (d) Selecting outlines using diffuse virtual lights at position 2 and 3. (e) Selecting shape lines using a diffuse virtual light at position 4. (f) Position of the virtual lights.*

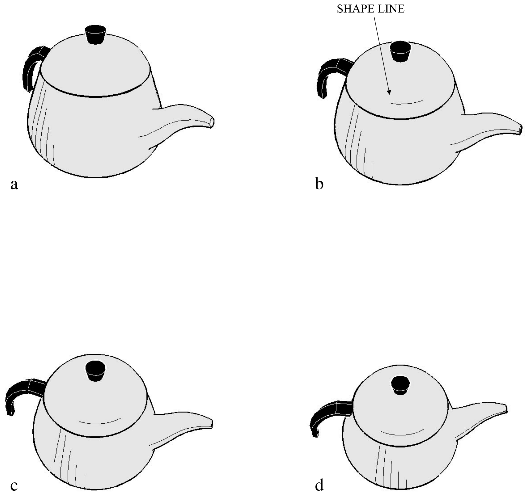

**Figure 14:** *Application of a specular virtual light to the teapot. Selecting shape lines using a specular virtual light at position 5. In (b), the shape lines appear. In (c) and (d), the shape lines change following the observer.*

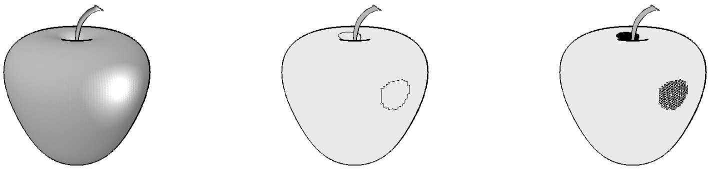

**Figure 15:** *Example of an apple showing a highlight and how it is rendered as a contour curve and as a fill area using silhouettes.*

**Figure 16:** *Example with a water pot, or "pipo", showing different modes of "artistic" rendering, changing the diffuse and specular virtual lights and the shading mode.*

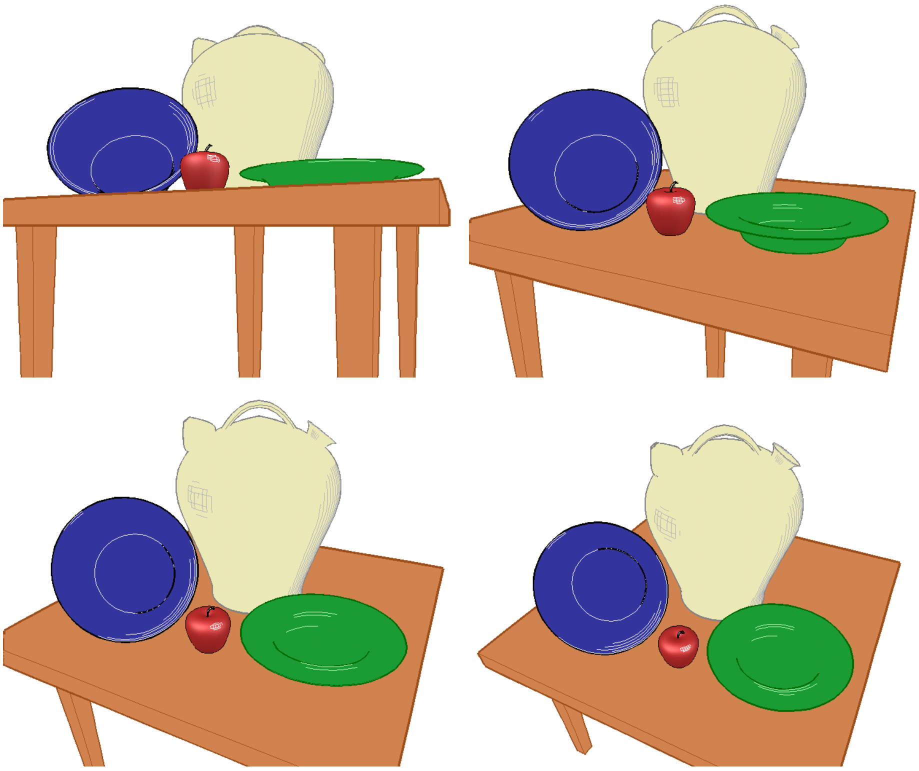

**Figure 17:** *An example with several objects applying different diffuse and specular virtual lights and shading.*