Flattening 3D objects using silhouettes

D. Martín, J.D. Fekete and J.C. Torres

Abstract
An important research area in non-photorealistic rendering is the obtention of silhouettes. There are many methods to do this using 3D models and raster structures, but these are limited in their ability to create stylised silhouettes while maintaining complete flexibility. These limitations do not exist in illustration, as each element is plane and the interaction between them can be eliminated by locating each one in a different layer. This is the approach presented in this paper: a 3D model is flattened into plane elements ordered in space, which allows the silhouettes to be drawn with total flexibility.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Line and Curve Generation

1. Introduction
In some applications, such as illustration, the user needs to transform and manipulate silhouettes in an artistic way, changing, for example, the width or the style, with no limitation. There are many methods that enable us to obtain silhouettes, from 3D models or from a raster structure, but they do not offer this level of flexibility. For example, if silhouettes are 3D curves obtained from a 3D model, the final result may be incorrect when such curves are very wide, because this effect produces a new geometry and relations between the components that are different from the original ones. Full flexibility is obtained when a 2D environment is used. In this case, there is no interrelation between the elements comprising each layer, and so there is total flexibility to carry out whatever the user wants, even if this produces something incoherent (of course, this is not necessarily bad, and sometimes it is called art).

Methods to obtain silhouettes can be classified by the dimension they are extracted from. Wikenbach and Salesin use a planar map to define where the silhouettes appear, their appearance and the placement of stroke textures. The same authors present a method for obtaining silhouettes from parametric surfaces, while in Salisbury’s work, the main goal is scale independence. The works of Markosian et al. and Gooch et al. present fast extraction algorithms. In Elber’s paper, free form polynomial and rational surfaces are used. Toledo’s method detects the outlines that are applied to textures in cell animation. Buchanan uses a data structure, the edge buffer. Another approach is to attach the silhouettes to a simple geometry, as used with grafals and their extension, the geografals. Hertzman and Zorin use the concept of a dual surface to find the silhouettes.

Saito and Takahachi use G-buffers (raster structures with 3D information) to obtain the silhouettes. The Piranesi system adds the material buffer. Deacaudin uses the z-buffer and also a buffer with the information of normals. The 2D method that is presented in works with stencil and frame buffers. Akeley’s method also uses the frame buffer. The work of Wang et al. solves the problem of drawing edges over the solid representation in OpenGL.

Martín’s work develops the Virtual Lights model, which allows the user to define when, how and where the silhouettes will appear in an object. There are also works concerning the appearance of the silhouettes. Sousa’s papers are mainly oriented towards the appearance of the silhouettes, simulating a graphite pen. Northrup allows a great variety of styles of silhouettes. Hsu’s work relies on 2D elements to produce different styles and forms.

The study that is most relevant to our work was that by Northrup. This solution is based on obtaining the projected silhouettes from a polygonal model, drawn on a plane in
Figure 1: Example of LPEs for a convex and a concave object. A) The convex object has only one LPE with a closed silhouette. B) The concave object has two LPEs, the first one with a closed silhouette, and the second one with two open silhouettes.

front of all the other components of the scene. This is a limitation because it may produce situations in which they interfere with each other when they are transformed, thus changing the form and style.

We present a new approach to the problem of the flexibility when rendering silhouettes: the flattening of a 3D scene to obtain a multilayer 2D environment. In other words, the objects are converted into 2D plane elements, which are ordered in depth. The key idea is to use the silhouettes to produce the plane elements, which are composed of an area part, represented by a polygon, and one or more linear parts, represented by the silhouettes. The form of the object can be transformed artistically, not only by changing the attributes of the silhouettes, but also those of the polygons (e.g. coupled with Meiers' work\textsuperscript{14}).

2. The algorithm

We first present an informal description of our method and how it is applied to convex, concave and composed objects. The main idea is that, given a 3D scene, each component can be converted into plane elements that are parallel to the projection plane, in such a way that the projected image of the 3D scene is equivalent to the composition of these 2D plane elements, like pushing the 3D objects until they are flat. For example, given a sphere, the plane element is a circle (for the moment, we do not take colour into account, just the shape). The plane representation is termed a \textit{layered plane element}, LPE.

Each LPE has two components: an area, represented by a closed polygon, and a closed silhouette (which can be divided into two or more open silhouettes in the latter stages of the method). For example, the LPE of the sphere, the circle, can be drawn as a circumference; the silhouette, as a filled circle without the border, the polygon, or both.

An object can be divided into one or more LPEs. A convex object always produces a LPE with one polygon and one silhouette, which is closed (Figure 1, A). A concave object can produce one or more LPEs, each of which can have one closed silhouette or several open ones (Figure 1, B). Once we have the LPEs of each object, they must be ordered. The correct image is obtained by drawing them from back to front.

The problem is how to produce the LPEs. The solution to this is based on using the silhouettes, the limit between the visible and invisible parts of an object. Expressed in an informal way, a LPE is the 2D representation of the part of an object that is limited by a closed silhouette. This part is usually, but not always, visible.

The scheme of the method is as follows:

\begin{verbatim}
for each object do
  Define and extract the silhouettes
  Obtain LPEs using connectivity and visibility information
  Divide and clean the LPEs to eliminate useless information
end for
\end{verbatim}

Once these steps are applied to all the objects, we have a set of LPEs. We must then order the LPEs and, from back to front, render each one (polygon + silhouettes).

Although, in our system, a style is applied to the polygon and to the silhouettes, it is possible to export the information to be used by an illustration program. In our current implementation, the models are 2-manifold polygonal ones, with no holes and no interpenetration. As our final goal is to produce "artistic" representations (stylised silhouettes, simple colour models, etc.) of 3D scenes, the restrictions are not very demanding.

2.1. Definition and extraction of silhouettes

The method is implemented using a polygonal model, based on a Winged-Edged data structure\textsuperscript{1}. This allows us to obtain the necessary topological information. We define and extract the silhouettes using the method presented in\textsuperscript{13}.

2.2. Creation of LPEs

One property of 2-manifold polygonal models is that all the edges that constitute the limit between the visible and invisible parts of an object, the silhouettes, are capable of producing a closed path, that is to say, starting from one point, a
chain of edges can be obtained that includes the same point as its end. This is very important because this characteristic allows us to define the LPEs. In fact, a LPE is obtained from the projection of a closed silhouette.

In order to obtain a LPE, we create a list of points from the list of edges marked as a silhouette. Each point has one or more chain paths associated. A chain path maintains the information from the previous edge and the next edge, which allows us to obtain the closed silhouettes. A point with only one chain path is called a simple point, while one with two or more chain paths is called a multiple point. The edges of a chain path are convex if the angle between the normals of both faces is greater than 180 degrees, or concave, otherwise.

2.2.1. Convex Objects

For a convex object there is only one closed silhouette that will produce a LPE. The information of the chain paths is used to create a chain of edges, the closed silhouette. To start the closed silhouette, the first point of the list is taken and marked as used. The previous marked edge will be the last one. The next point can be obtained from the next marked edge and marked as used, and so on until the first point is reached. For convex objects, all the points are simple, and all the edges are convex.

2.2.2. Concave Objects

Concave objects can produce complex configurations with one or more closed silhouettes. For example, in Figure 2, A, although there are several closed silhouettes, they can be obtained in the same way as with convex objects. The points of the list are taken to form silhouettes until all the points are marked as used. In Figure 2, B, there is only one closed silhouette, but when it is projected the polygon is not simple. The LPE has only one polygon and two open silhouettes. The polygon is the convex hull of the projected silhouette, while the closed silhouette is divided, in the final stages of the algorithm, into two open silhouettes, to produce a correct visual result (Figure 8). The most complex case is shown in Figure 2, C: here there is no clear division between closed silhouettes, as there are several points where the silhouettes converge, the multiple points. The problem is to obtain the chain paths that will produce the correct result.

We mark a point as multiple if more than two convex silhouette edges converge to it (e.g., the object in Figure 2, C, is represented as in Figure 3, A). In such a case, there are also concave silhouette edges that converge. If all these edges are projected and a direction is selected (for example, counter clockwise), it is possible to define the two edges that separate the inner zone from the outer one (Figure 3, B). Given

Figure 2: Different configurations of silhouettes from which the cell layers are obtained. A) Two LPEs with closed silhouettes and only simple points, B) One LPE with two open silhouettes. C) Two LPEs with closed silhouettes and multiple points.

Figure 3: A) Representation of a concave object (from silhouettes of Figure 2, C). B) Elements of a multiple point. C) The chain paths are created following the direction from the first edge to the second edge by the inner zone.

Figure 4: Two possibilities to produce the LPEs. A) Using the rule. B) Creating a back LPE.
Figure 5: In some configurations, the silhouettes can enclose invisible zones. B) The object is divided into two silhouettes. C) The LPE of the second silhouette must only draw the convex border.

In some cases the correct silhouette cannot be obtained. In Figure 5, B, two paths start from the multiple point but none produces the correct result. The problem is that with the paths it is impossible to produce a closed silhouette in which all the faces that are included are only visible or invisible. Instead, we can produce an outer silhouette that includes visible faces, and an inner silhouette that includes invisible ones. In this last case, the LPE has an open silhouette, corresponding to the part where the edges are convex (Figure 5, C). To resolve this case, a third edge is added to the chain path; this is used to attempt a second silhouette if the first fails.

Although this algorithm normally produces a good LPE, the result may sometimes be incorrect. The problem is not with the silhouette part of the LPE, but with the polygon, if the latter does not cover the other parts properly. In Figure 6, A, each LPE is drawn with a different colour. The silhouettes, together with the polygons filled with the same colour, are shown in Figure 6, B.

2.2.3. Composed objects

The method has been extended to composed objects, even with parts that interpenetrate (Figure 7, A). In this case, if we process the scene component by component, the LPEs would be wrong, because they represent the complete structure, without taking the inclusion into account. The solution consists of computing the intersection of one piece with another, and marking it as forbidden to draw.

Another possibility is to provide the user with the capability of manually simulating the intersection, marking the geometry in such a way that there are both normal points and clip points. This attribute is used in the division stage, as explained below.

2.3. Division and cleaning of LPEs

At this stage, the object has been divided into one or more LPEs, with a polygon and a closed silhouette.

Sometimes it is necessary to divide a closed silhouette into several open ones. This is done by using information about the type of edge (convex or concave). Both types indicate a
change in the visibility, but given the direction of the camera, a convex edge represents a silhouette in which the nearer face is visible and the other one is invisible. The opposite is true for a concave edge. Each chain path has information about the edge type of the previous and the next edge in the chain. Four cases are found: convex-convex, convex-concave, concave-convex, concave-concave. The rule we use is that, in the CCW direction, an open chain begins at a concave-convex point and finishes at a convex-concave one (Figure 8, B). Using this rule, a closed silhouette can be divided into several open silhouettes, associated to the same polygon. At the same time that the open silhouettes are obtained, it is also necessary to compute the intersections between them, to eliminate the invisible parts (Figure 8, C).

This same process can be used to produce a manual intersection. In the same way as previously explained, we search for changes in the type of point: from normal to clip, and from clip to normal (Figure 9, A). The edges between two clip points are eliminated (Figure 9, B). It is possible to find several changes, which produce new chains. This solution is capable of producing good results, with little effort.

The last step is the elimination of duplicates, that is, those points that are consecutive and have the same projection. This is done because it is necessary for all segments to have a length of at least one pixel for the creation of the strokes, which are used to draw the silhouettes.

2.4. Ordering

The LPEs must be ordered in such a way that when they are drawn, from back to front, the result obtained is visually correct, that is to say, the depth relationship between all the LPEs is visually coherent.

In most cases, we cannot use the depth information obtained from the silhouettes to compute the position of the LPEs directly, because they do not provide a unique value. Instead, it is possible to determine the relative position of one LPE (the polygon) in relation to another by analyzing the inclusion relationship between them. There are three possibilities: a) there is no contact between LPEs; b) one LPE intersects with the other; and c) one LPE is included in the other.

The first case presents no problem: here, the order is immaterial. In the second case, the order is established by computing one intersection and testing the depth value. When one LPE is included in the other, the solution is based on the visibility of the points: if they are visible, the included LPE is in front of the other one; if they are invisible, the LPE is occluded and must be eliminated.

There are several ways to compute the visibility of the points. We use the method based on frame-buffer identification. This approach presents some problems due to aliasing, but as we are not interested in images with fine details, the resolution of the polygonal models can be simplified without adversely affecting the result. The relationship of all LPEs is computed in a structured way. A binary tree is used, and this is recreated each time a polygon is included. Finally, the preorder result of the binary tree gives the correct sequence of LPEs.

Note that the visibility is computed at LPE level. So, if one object is at the same time both ahead of and behind another object, sometimes it will be drawn ahead and sometimes behind, depending on the first intersection computed, and the algorithm may not provide the expected result. If this is a problem (for instance, to do animation), the object can be divided into two or more parts that can be ordered as in the previous scheme (Figure 7, C). The new geometry that appears must be marked indicating that it may be part of a silhouette, but that it is forbidden to draw it.

2.5. Rendering

The final step is to draw each LPE. We are interested in producing "artistic" representations. With LPEs, it is possible to work with the silhouette or silhouettes, the polygon, or with both. For example, we have created some new effects of the broad palette of 2D rendering styles that are used by artists.

Our main goal with this project was to produce special effects with the silhouettes. To obtain the required flexibility, and wide lines, especially, we used the same method as
For the polygonal part, we have developed a method that attempts to produce a “cubism” effect. Given the polygon, the idea is to simplify the number of points, in such a way that the simplified version is related to the original one. The method computes the angles between segments, which are then ordered. Finally, the points with the smallest angles are deleted. The user specifies the percentage of points to be deleted and the number of steps in which this is to be done. This simplification process can also be applied to the silhouettes. To produce the final cubism effect, the simplified polygon is randomly scaled.

It should be noted that the LPE method enables us to produce many more effects, some on the silhouettes and some on the polygons. For example, it is a simple matter to produce an image with a realistic colour model for the polygons and to add the information of stylised silhouettes. One possibility is for each object to compute the realistic image, which is then clipped using the polygons of the LPEs as masks. To obtain the final result, these are drawn instead of the polygons.

3. Results

We provide several examples to show the capabilities of the method, although the most important is probably the fact that LPEs can be exported to illustration programs. The program is implemented in C++, using Mesa3D and Linux. We have used a Pentium II 266 Mhz, with 256MB of RAM to obtain the results. The performance varies from 1 to 0.5 fps in the examples commented below.

The example in Figure 12 shows the possibility of making wide silhouettes without any interactions. It is possible to produce very wide lines, although the result may be strange. Figure 13 illustrates a style that simulates pen sketches applied only to the silhouettes, with the silhouettes drawn in white, and with coloured polygons. The examples in Figure 14 show different possibilities when points are used. The airbrush effect is simulated with points by changing the width of the silhouettes.

Given that the polygons and the silhouettes of a LPE are independent, we can apply different effects to each. In Figure 15, the still life is drawn in a “cubism” style (simplification+scaling), while the silhouettes have a displacement and thin-extremity effect. Another way to avoid a mechanical appearance in silhouettes is by applying displacement functions. The two still lives are given the same effect, by just changing some parameters (such as the amplitude or the phase of the sinus function). The extreme points are also thinner (Figure 16).

The example in Figure 17 shows a model of a water-pot drawn with different styles. Figure 18, shows another model. Different effects are applied to the chair. From top to bottom, we see the wire chair, the chair with the normal silhouettes but with the extreme points thinner, the same effect as the
previous one, but with an added displacement (a random sinus function), the “cubism” chair, the chair with a style that simulates a charcoal effect by using points, an airbrush style based on wide strokes, a sketch style, using parallel lines, with very wide strokes, a sketch style using only two lines with thin strokes, a furry style using radial lines, and finally, two sketch styles using perpendicular lines.

In Figure 19, only the polygons are drawn, without silhouettes, without effects and with the “cubism” effect.

We have produced some short animations, but depending on the style, the coherence between frames is not total; sometimes, though, this is an effect sought by the user.

4. Conclusions

Silhouettes are important in some areas of non-photorealistic rendering. In many applications, they are used as an aesthetic component, which must be stylised to produce artistic results. This approach is achieved by means of variable-width lines, by changes in style, etc. This goal is normally achieved by producing 3D components (curves, triangle strips, etc.), a solution that limits flexibility due to the restriction of the spatial relationship between 3D objects. A step forward is to provide all silhouettes in a plane, in front of the entire scene, but the full flexibility used by illustrators is only found in 2D systems. In this case, the layers eliminate the relationship between components and the user can do whatever s/he wants.

Here, we present a method for “flattening” 3D objects, thus obtaining 2D elements, the LPEs, which are composed of an area part, a polygon, and a silhouette part. The method in its current implementation has some limitations, but nevertheless opens new possibilities for artists, establishing a bridge between the worlds of 3D and 2D.

As future work, we are developing a method based on spatial division to compute the visibility, and different methods of obtaining other feature lines and silhouettes in complex configurations of multiple points.

Acknowledgements

The authors thank the referees for their useful comments and suggestions that have contributed to improve this paper. We are grateful to the PaVR European TMR Project, to the Ecole des Mines de Nantes and to the Departement Informatique.

References

Figure 12: Example of wide silhouettes without style.

Figure 13: Example of a style based on pen sketches, applied to the bodega, without the polygons and with colored polygons.

Figure 14: Example of a style based on points, changing the width of the stroke.


Figure 15: Simulation of a "cubism" paint (see colour section).

Figure 16: Application of displacement functions to simulate hand-made sketches (see colour section).

Figure 17: Example of a "pipo" with different styles (see colour section).
Figure 18: Application of different styles to a chair (see colour section).

Figure 19: Using only the polygonal part of the LPEs, without and with an effect applied (see colour section).