SymSketch: Modeling 3D Curves using Sketching and Symmetry

SUBMISSION ID : paper1251

Abstract

We present SymSketch, a novel sketch based interface for the conceptual design of symmetric 3D objects. Our approach uses the insight that a pair of symmetric 3D points can be uniquely constructed from their 2D projections in an arbitrary camera view. SymSketch users sketch strokes freely in an interactively controlled camera view and plane of reflection. A newly sketched stroke is compared against existing strokes for its reflected projection in the current view and a matched symmetric pair of 2D strokes are automatically turned into 3D curves. We also construct object topology by connecting strokes to proximal existing curves, creating 3D curve networks suitable for surfacing. SymSketch interactively guides users towards good camera views for sketching using a through-the-lens visualization of view quality. Reflection planes are controlled using a novel three point manipulation widget. We further integrate curve editing, deletion, and asymmetric curve creation into a fluid sketch based work-flow. Informal user evaluations of SymSketch show it to be a simple and appealing interface for the conceptual design of symmetric 3D objects.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Modeling packages I.3.6 [Computer Graphics]: Interaction techniques

1. Introduction

Research in conceptual modeling continues to battle the challenging question: How can a designer quickly and effectively transform a mental concept into a digital object that is easy to refine and reuse? Sketching, while clearly the most common technique for early stages of 3D conceptual design, is inherently a 2D process. The ambiguity between a sketch and its intended 3D structure needs to be resolved either explicitly by the designer or implicitly based on knowledge of the object being designed. In this paper we automatically infer the 3D shape of objects by exploiting their symmetric structure.

Symmetry in nature is manifested in the form and function of living organisms. Neuroscience suggests that the concept of symmetry is hardwired into our visual processing system and influences our perceptual understanding of shape [NCP+02]. Consequently, symmetry is an important principle which guides the aesthetics and construction of synthetic objects [Arn54, Gom69, Wey52]. The fact that symmetry is ubiquitous in design is reaffirmed by the functionality to symmetrically replicate objects commonly found in most commercial and research modeling software [TBSR04]. We go a step further in that we also use the knowledge of symmetry to aid in the construction of 3D objects from 2D input.

The goal of our system, SymSketch, is to produce a collection of 3D curves, that can be used as an aid to understanding volume, and as input to many downstream design applications. We exploit symmetry to construct curves in 3D from sketched pairs of 2D strokes. We build upon the insight that a single view of a bilaterally symmetric object can be conceived as two views, the second resulting from reflecting the current camera with respect to the plane of symmetry [FMW03]. A pair of sketched 2D points that are known to be symmetric with respect to a given 3D reflection plane can thus be triangulated uniquely into 3D points (see Figure 1). Connecting such point pairs yields curves in 3D.

The user work-flow for SymSketch is as follows (see Figure 2): A user interactively defines views in which to sketch using the ground plane and plane of symmetry as a visual guide. The user then sketches strokes which are images of 3D curves in this view (see Figure 2a). When images of symmetrical curves are detected, SymSketch automatically uses curve symmetry and constraints of topology to con-
Figure 2: SymSketch Workflow: Matched pairs of 2D strokes drawn in a given view (a) are turned into 3D curves (b). Unstable regions for sketching are shown in red (c). A sym-tree of planes and template geometry provide a design scaffolding (d) and a 3 point widget manipulates the current reflection plane (e). The end result of SymSketch is a network of 3D curves (f).

2. Related Work

Our research draws primarily from the areas of sketch based modeling and computer vision. Sketch based modeling provides us with a body of work surrounding the task of creating interfaces and workflows suitable for pen-based interaction. From computer vision, we draw the mathematical techniques which allow the symmetry based reconstruction of three dimensional curves.

2.1. Sketch Based Modeling

The objective of sketch based modeling systems is to generate three dimensional geometry through sketching and sketch-like metaphors. The SKETCH [ZHH96] system uses gestures to create, shape and locate primitives such as rectangular solids and cones. Teddy [IMT99] uses a combination of sketched strokes and gestures to model blobby objects. ShapeShop [SWSJ05] and SmoothSketch [KH06] extend this paradigm and can create more complex objects.

Work which focuses on curve creation has produced a wide variety of methods by which the three dimensional depth ambiguity may be resolved. One approach is to intersect the outgoing ray with an existing 3D element, such as a plane [Kal05, TBSR04] or a template mesh [KS07]. Because the curves are constrained to existing geometry, these systems do not allow the creation of arbitrary shape and must ameliorate this through modification mechanisms [KS07], suggestions [TBSR04], or moving the target geometry during the course of the stroke [Kal05].
Two previous systems used symmetry constraints to facilitate 3D curve creation. [BKK03] presents an interface in which a user can model a symmetric pair of degree cubic curves through sketched intersection with a “see-through box”. Kanai et al. ([KFT92]) present a system which models cubic Beziers from symmetrical curve pairs. The use of splines makes incremental curve editing and the incorporation of arbitrary design constraints from template geometry, images and other design collateral [TBSR04, KS07] difficult. We formulate the symmetric matching problem along the lines of more recent research, using piecewise linear curves within a general energy minimization framework subject to various design constraints [KWT88, TBSR04, KS07]. These systems do not explicitly address workflow issues of interactive control, planar hierarchies and unstable sketching views in the context of a conceptual sketch-based design system.

Other types of constraints have also been applied to depth resolution. Sketch3d [MSK02] matches the sketch to a 3D template curve network of known structure. This approach is powerful, but is limited to sketches of deformed versions of the 3D template. Work by Das et al. [DDGG05] resolves the depth ambiguity by minimizing curvature among curves lying in the possible set corresponding to a given stroke.

Our system is closest in spirit to the work of Karpenko [KHR04] and Cohen [CMZ’99] which generate true free-form curves by leveraging multiple curve representations. Karpenko allows strokes to be sketched in multiple views selected by the user while providing epipolar cues to the ambiguity remaining after prior strokes. Because our system allows geometry to be created from a single view its work-flow is closer to that of natural sketching. The system proposed by Cohen does allow curves to be modeled in a single view by sketching both the curve and its shadow on the ground plane. The fixed ground plane restricts the shape of the generated curves and can force the artist to draw from undesirable viewpoints.

2.2. Computer Vision

The basis of our reconstruction method is drawn from findings in computer vision. Mundy [MZ94] discusses how bilateral symmetry and other repeated structures can be applied to the problem of 3D structure recovery. François [FMW03] provides a formal proof of the equivalence between a single image of a symmetric model and two separate images of the same object. He then uses this equivalence to reconstruct the 3D structure of meshes of known topology by manually providing a correspondence between symmetrical mesh vertices.

Recent work by Hong [HMY’04] applies symmetry to the reconstruction of three dimensional curves. This work includes an elegant solution to the symmetric point-pair reconstruction problem which performs well even when the points in question are imaged incorrectly. An automatic method is presented by which points on the curve interiors can be paired with their symmetric opposites, which allows for the point-wise reconstruction of 3D curves. We use Hong’s point-pair reconstruction technique, but the correspondence method he presents requires strokes to be more precise than those generated by sketch artists.

3. 3D Reconstruction Using Symmetry

We now detail our formulation of 3D curve construction from a sketched pair of symmetric strokes. We first formally define the camera and reflection plane used in our approach.

3.1. Camera

The projective camera is represented by a matrix which projects three dimensional world space points onto a two dimensional image plane according to the relationship PX = x, where P is the 3 × 4 projection matrix, X is a 3D homogeneous world space point, and x is the 2D homogeneous image space projection of X.

The matrix P can be decomposed as P = K[I|0]V, where K is a 3 × 3 matrix which represents the camera’s internal parameters and V is a 4 × 4 “view matrix”, which transforms world space coordinates into the local view frame of the camera.

In this paper, we consider natural cameras defined by orthonormal view frames, and square, non-skewed pixels. For a natural camera, we have

\[ K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & -1 \end{bmatrix} \]

, where f is the camera’s focal length. The -1 in the bottom right element of the matrix reflects a shift from a right handed view coordinate system to a left handed image space coordinate system.

Since the natural camera’s view frame is orthonormal, we can also decompose V as V = RT, where R is a pure rotation and T is a pure translation.

The eye-point is set to the origin of the view frame. The camera’s world space location C in homogeneous coordinates is thus \[ C = V^{-1}[001]^T \].

Since camera projection maps 3D space to a 2D plane, an image space point x does not correspond to a unique world space point X but instead to any 3D point along a line l given by

\[ X = \tilde{C} + \lambda(P^+x - \tilde{C}) \] (1)

where \( P^+ \) is the pseudo-inverse of the projection matrix, and \( \tilde{C} \) is the Cartesian equivalent of C.
3.2. Bilateral Symmetry

Bilateral symmetry is defined as an operator which reflects space about a plane \( p \). In matrix form, this operation can be described as \( S = Z^T \), \( RZ \) where \( Z \) is any rigid matrix which maps \( p \) onto the x-y plane and

\[
R = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & -1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

is a matrix which flips the z-axis of the coordinate system.

The operator \( S \) is a one-to-one operator which maps world space points \( X \) to their symmetric opposites \( X' = SX \). A model is said to exhibit bilateral symmetry if for every model point \( X_i \) its symmetric opposite \( X'_i \) is also a point in the model.

3.3. Point Pair Reconstruction

A pair of world space points \( X \) and \( X_r \) are symmetric under a symmetry \( S \) if \( X_r = SX \). These points are projected by a camera having matrix \( P \) onto image points \( x \) and \( x_r \). We consider the view to consist of the set of camera parameters which control a perspective projection of features in three dimensions to the screen. These parameters can be classified as \( external \) and \( internal \). The external parameters are the physical parameters which situate the three dimensional camera in space: the camera’s location, view vector, up vector, and eye vector. The internal parameters control the inner details of the projection, such as pixel aspect and skew, station point (the location on the image plane being directly “looked at”), and focal length.

In our interface, we consider only so-called “natural views”, where pixels are square, non-skewed, and the external camera vectors are orthogonal. This is no limitation, as almost all views found in sketches and photographs exhibit these properties.

4. Interaction Techniques

We thus use a two step approach for curve reconstruction within a general curve editing framework that minimizes constraint energy of 3D curves. The first step in this approach is to generate an approximate correspondence between points on the two strokes. We assume strokes in our system to represent generally smooth curves and preprocess a stroke with sharp tangential discontinuities into multiple connected smooth strokes. We then densely resample each stroke into equal arc-length segments so that each resampled stroke contains the same number of points. This approximate correspondence is then used to generate an initial guess at the 3D curve’s structure. We then refine the curve structure using energy minimizing snakes [KWT88]. The curve endpoints are fixed, while the curve interiors are free. The symmetric energy term for a 3D curve point is the point-to-curve shortest distance in 2D between the point projected back through the camera and the sketched stroke. In addition to this term we employ an energy term for smoothing [TBSR04] and optionally, terms for other design constraints [KS07].

4.1. View Control

The key ingredients for a sketch based symmetric object design system are the camera, reflection planes and of course stroke sketching for curve creation and editing. Our systems adheres to many of the design guidelines laid down by prior sketch based modeling systems [IMT99, SWSJ05] and in this section we now focus on the specific interaction innovations motivated by the design of symmetric objects.
As an indication of the current view properties, our interface shows the user an image of the ground plane with overlaid central axes. In addition, the current plane of symmetry is indicated as a featureless transparent rectangle. These cues serve as a visual guide to situate the designer in the interface’s three dimensional world.

External parameters are modified through typical tumble and dolly actions. Focal length, the only salient internal parameter, can also be changed interactively. These controls are accessed using a keyboard modifier. While this style of “clutch and drag” interaction may not be the ideal fit for a pen-based interface, the frequency with which these operations are accessed makes this a good choice [TBSR04].

Ironically, SymSketch strives towards single view sketching, with strokes being resolved from 2D as 3D curves as and when symmetric matches are detected. In this regard unresolved strokes need to be deleted when the view is altered. While camera manipulation in SymSketch is less frequent than other 3D modeling systems artists often manipulate the view to get a slightly different look at what they have created. To facilitate such viewing and to safeguard unresolved strokes from accidental deletion, these strokes start to interactively fade out as the camera deviates from the original view. If the user releases the camera while the strokes are still visible, it animates back to its original view and the strokes persist, otherwise the new view remains and the strokes are deleted.

4.2. Stability Display

The heart of the process by which the 3D curve location is determined relies on the intersection of two rays in space. When these rays become nearly parallel a situation arises in which small errors in the sketch are magnified, creating large errors in world-space. Experienced artists avoid these situations naturally by intuitively choosing views which are most expressive of the 3D model structure.

To help less experienced artists develop this intuition, SymSketch provides an approximate view of areas in which the reconstruction will be less robust against sketching errors. When this display is active dangerous regions are highlighted in red. As can be seen in Figure 3, views in which the camera lies close to the symmetry plane and views which face the plane directly naturally provide less 3D information.

4.3. Symmetric Curve Drawing

Drawing strokes to indicate curves is the central modeling operation of our interface. The strokes represent screen projections of curves in the desired model. Black sweeps are used to represent strokes which do not yet exist as three dimensional objects.

When enough information exists to “move” these strokes into 3D, reconstruction is performed and the three dimensional curves are displayed in blue and with a subtle shading effect to indicate distance from the camera (Figure 4).

There are three separate cases in which enough information exists to calculate symmetric 3D curves from strokes:

Symmetric curve pairs: (Figure 4) The most general situation in which 3D reconstruction may be performed occurs when two strokes are sketched which represent the projected images of a symmetric pair of world space curves.
We require that these curves be sketched in their full form without any hidden line removal due to occlusion. When a stroke is drawn it is matched against all currently unresolved strokes. Candidate strokes for matching are first culled by looking at the epipolar relationship of stroke endpoints and then more comprehensively matched as described in Section 3. The best match with an fitting error below a given threshold resolves the strokes and the corresponding reconstructed 3D curve pair is inserted into the model. If no stroke matches the current stroke it remains unresolved to be potentially matched by a subsequent stroke.

**Self-symmetric curves:** (Figure 5) The user indicates a self-symmetric curve by drawing a stroke which connects the endpoints of a pair of symmetric curves. When this situation is detected a modified version of the reconstruction is performed which is guaranteed to produce a single world space curve with self-symmetry. The new curve is then glued to the curves it connects using the techniques described below.

It is interesting to note that the effect of self-symmetry can be approximated by sketching a stroke and then over-sketching it in the reverse direction. This will produce a closely aligned pair of world space curves with near self-symmetry. The single stroke method described above provides a shorthand for this operation. In addition, it has the advantage of generating a single world space curve with true self-symmetry.

**Curves on the symmetry plane:** (Figure 6) A similar technique is provided to allow sketching curves which lie on the symmetry plane. This interaction is triggered when a stroke is sketched which connects two existing curve points which lie on the plane of symmetry. These points can either be curve endpoints or midpoints of self-symmetric curves. In this case, a single curve is generated lying on the symmetry plane, which is then glued to the constraint points in world space.

**4.4. Topology Enforcement**

It is often the case that the object being modeled can be represented by a connected network of curves with a well-defined topology. Such a network is useful as input to a process which fits surfaces to the curve network. In these situations the model which is generated should reflect this constraint by being “water-tight”, with adjacent curves having endpoints which are welded together.

Our interface provides support for this type of curve network by enforcing topology through endpoint snapping. When a stroke is drawn which has an endpoint lying close to the projected endpoint of an existing curve, a constraint is formed which will connect the reconstructed curve to the existing curve. It is important to note that these constraints are detected and formed in two dimensional stroke space, but enforced in three dimensional curve space.

The combinations of these constraints lead to two distinct situations, which are handled differently:

- If only one endpoint is constrained, a simple translation is applied to the reconstructed curve to glue that endpoint to the existing model curve.
- If both endpoints are constrained, a combination of translation, rotation, and uniform scaling are applied to glue both endpoints into position. The use of this rigid transformation serves to preserve the character of the new curve.

In cases when a self-symmetric curve has a sharp discontinuity on the symmetry plane, users may draw the curve as two strokes with a common end-point on the symmetry plane. In this case these end points after reconstruction in 3D are precisely projected to a 3D common end point on the reflection plane. This common 3D point is then used for the remainder of the 3D curve construction, ensuring that the new 3D curves meet at a single point on the symmetry plane.

**4.5. Curve Editing**

When sketching, designers typically do not erase unsatisfactory strokes, but rather repeatedly sketch over these strokes until the desired shape is achieved. In analogy to this technique, we provide a curve editing facility based on over-sketching the images of 3D curves (see Figure 7). A stroke is recognized as an edit when two conditions are satisfied:
Both of the stroke’s endpoints lie close in the image to the projection of the same 3D curve
• The tangent of the projected curve is closely aligned to that of the stroke at both of the stroke’s endpoints

After the stroke has been recognized as an edit, a new 2D stroke is generated by splicing the modification stroke into the 2D projection of the curve to be edited. The proximity and tangency constraints ensure that this can be done without creating strokes which are unreasonably jagged. Reconstruction is then performed using this merged stroke as if it was a curve creation stroke. In the self-symmetric and on-plane cases, no other information is necessary to preform the reconstruction. In the case in which the edited curve is a member of a symmetric pair, reconstruction is performed using the merged stroke and the projection of the symmetric opposite of the curve to be edited.

4.6. Planar Curves

The ability to model free-form curves in 3D is one of the strengths of our interface. Nonetheless, situations often arise where planar curves are necessary. Planarity can be encouraged by adding an energy minimization term that attracts points towards their projection on a best-fit plane. Alternatively SymSketch users can force the modeled curves to be precisely planar. In this case the constructed 3D curve is projected to a least-squares best fit plane containing the curve end points and then mirrored to create its symmetric counterpart.

4.7. Curve Deletion and Asymmetry

Curves which have been modeled in SymSketch can be deleted by crossing them with a short, straight stroke. As we have mentioned, symmetry is enormously prevalent in both the natural and the man-made worlds. Despite this fact, many models contain asymmetric features. Though the most obvious use of SymSketch is to model symmetrical objects, it can also be used to create features which are asymmetrical. Here the mental model is not of sketching a single object with symmetry, but rather that of sketching a single object and also its image in a mirror. This work-flow creates two mirrored copies of the asymmetrical object. The curves “in the mirror” can be deleted when they are no longer needed for modeling.

It is also possible to directly model asymmetric curves through a projective technique. After a user draws a stroke which connects the endpoints of two existing curves, he may choose to “force” the stroke into 3D. In this case, the 3D generated will be a simple planar projection of the sketched stroke. The plane onto which the curve is projected is that which contains both of the connected endpoints, and also the up vector of the current camera. This technique is limited in that it produces only planar geometry, but provides a means to generate asymmetry with a single stroke.

4.8. 3D Templates

In many cases it can be helpful to have a reference point for the design at hand. This reference may take the form of an associated object such as a hand which holds a mobile phone, a set of engineering criteria for the design of an automobile, or an earlier incarnation or iteration of the design (commonplace in the automotive industry). These references can be used to guide the design by providing scale, context, and other cues.

In our interface, we allow the user to load a three dimensional model to be used as a “design template”. Such a template is shown in context with the other UI elements, and serves as a visual reference for the modeling, as seen in Figure 8. In the future one can easily imagine enhancing

Figure 7: Curves can be modified by over-sketching

Figure 8: Template geometry serves as a visual reference
4.9. Symmetry Plane Control

Complex models often exhibit many planes of symmetry making symmetry plane manipulation an important system task. Standard translation arrows and arc ball widgets are not well-suited to sketch-based modeling and we thus develop a new 3-point widget suited to the task of planar manipulation.

The widget consisting of three drag-balls. Each ball defines a single point in space, which together define the unique symmetry plane. The balls are drawn in bright red as clear UI elements which are not part of the sketching or the 3D model. When occluded by the template geometry, the balls are drawn more faintly using a wire-frame style.

The balls can be moved by dragging them on the template geometry, or in its absence, constrained to the principal planes (Figure 9). This provides a simple way to define planes of symmetry in context and relevant to features of the template being used as a reference. It is sometimes useful to translate a symmetry plane along its normal without changing its orientation. When this is the case, the operation may be performed by dragging a ball while applying a keyboard modifier. The dragged ball moves as in the unmodified case, but now the other balls maintain their relative positions to the dragged ball, producing the desired effect.

4.10. Multiple Symmetry Planes

When models contain multiple planes of symmetry, be they local or global, it may become onerous to continually shift the symmetry plane to reflect the features one is currently modeling. SymSketch allows a fixed set of symmetry planes to be loaded if desired. It is possible to choose a current plane of symmetry by sketching a short stroke across the boundary of the desired plane. The selected plane will be used for further modeling operations.

We also allow the user to preload planes defined in a hierarchical tree. Curves which are modeled relative to a plane which is in such a tree are mirrored not only about the current plane, but also recursively about all of the plane’s ancestors in the hierarchy (Figure 10).

Hierarchical symmetry planes are especially interesting when used in conjunction with recent work on local symmetry detection in meshes [SKS06, PSG∗06, MGP06]. This work provides a method to detect a hierarchy of local symmetries which exist in a mesh. This hierarchy can then be loaded directly into SymSketch, along with the mesh from which it was extracted (Figure 11). This provides a means by which SymSketch can be used to extend and augment existing meshes containing symmetry.

5. System Hardware

Our system has been implemented on an Intel Pentium M 2.13 GHz processor. The graphical display is accomplished using the OpenGL API, running on an NVIDIA GeForce 6800 GO GPU. The primary input device used is a Wacom Intuos 2 digitizing tablet. On this system, all of the algorithms we employ run at interactive speeds. The interface
is not very demanding of the system, and has been tested on a variety of other processors and GPUs, always achieving interactive rates. While a digitizing tablet is preferred as a sketching input device, we are not currently using any of the tablet specific features such as pressure and tilt. Thus, a mouse or touch-pad can also be used as an input device.

6. Discussion

The aim of the SymSketch interface is to provide a bridge between the natural, fluid process of sketching and the precise, engineered process of 3D modeling.

To determine whether we achieved this goal, we performed an informal user study with both amateurs and professional designers. Users were asked to become familiar with an early version of our software, and to provide us with both models they had created as well as verbal feedback, to answer the following questions:

**Does the interface preserve the feel of sketching?**
Every user who was asked this question answered that modeling in SymSketch felt closer to sketching than to CAD-style 3D modeling. Users experienced in working with 3D modeling systems initially tended to manipulate the camera more but quickly got comfortable with a more persistent view style of modeling. Skilled artists but novice computer users were able to work efficiently without being confounded by the concept of a 3D virtual camera. The quick look to a nearby view without losing the current camera view or unresolved strokes was appreciated. Users also found changing camera views to be a nifty way to flushing out large numbers of unresolved spurious strokes (we only support explicit stroke deletion one at a time with a crossing stroke).

**Does the approach generate desired shapes?**
Even users who had little sketching experience were able to model curves well after just a few minutes of using the application. This was underscored by the complex shapes of the modeled objects. Often the shapes were what they wanted and were otherwise usually fixed by over-sketching. Some users found the topological snapping in SymSketch to be too aggressive, and sometimes prevented them from achieving the desired curves.

**Was the lack of tools supporting asymmetry limiting?**
As can be seen in Figure 12, symmetry was shown to be a useful tool for modeling a great variety of shapes. In fact out of all of the data we received from our users, no two models depicted the same type of object. None of the users mentioned feeling limited by symmetry. One of the users mentioned the application as being a wonderful way of exploring free-form space within the confines of symmetry and asymmetry could be thought of as a design refinement.

**Is curve quality acceptable for downstream use?**
To answer this question, we asked a 3D modeler to create surfaces directly from a network of curves generated by one of our sample users. The quality of the strokes (which were modeled by a professional designer) was such that it was possible to surface them using automatic tools included with an industry standard 3D surfacing package. The resulting high quality surfaces can be seen in Figure 13, in context with the original SymSketch model.

7. Conclusion

Overall, our system has demonstrated that symmetry is an effective means to facilitate sketch based 3D modeling. The strengths of our approach lie in symmetry’s ubiquity in both design and nature, and the ability to determine its 3D structure from a single 2D sketch. Users with a broad range of experience levels were able to use our system to create a variety of models of good geometric quality suitable for use in downstream applications. None of the users commented on the interface presented barring the success of the 3-point widget, which we take as a positive affirmation of our design decisions. Future directions include more user control over the topology enforcement mechanism, and the consideration of other aspects of the visual design process such as shading and highlighting.

Figure 12: Designs produced by our sample users

Figure 13: Curve networks authored in SymSketch were of sufficient quality to be quickly surfaced by commodity 3D software

submitted to EUROGRAPHICS 2008.
References


