On the Development of a Fully Automatic Methodology to Create Smooth Mesh for 3D Models from Concept Arts

Renato D. Viana, Erickson R. Nascimento, Renato A. C. Ferreira
Department of Computer Science — Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, Brazil

Abstract

In this paper we present a fully automatic methodology which creates smooth meshes for 3D models from conceptual sketches. Our approach is capable of generating meshes of different number of polygon without decreasing the quality. Traditionally, 3D models are manually produced by a 3D modeler, based upon sketches produced by a 2D artist. Although high quality models may be created by this process, it is slow and costly. Our solution receives conceptual arts of an object in different visualizations as input, and produces a 3D model as output. It is based upon known algorithms for capturing volumetric information from multiple 2D visualizations, and then converting it into a standard border representation. In this work, we have introduced algorithms and strategies to improve the quality of the model as well as its size, a compromise which is very difficult to balance. With our approach, we are able to generate relatively high quality models from several objects at very low cost. We verified the quality of the 3D models generated by our solution by comparing the models generated from synthesized views of manually produced models.

Keywords: Computer Graphic, Sketching, Modeling

Author's Contact:

{renato.viana,erickson,renato}@dcc.ufmg.br

1 Introduction

The process of designing and producing sophisticated virtual scenes, with many independent, three-dimensional objects, rich in visual and geometric details, is a very complex task. In general, to tackle with this task, the first step is to create a sketch of the general idea, a simplified drawing of the object of interest, for instance, a two dimensional illustration. This technique is called *Sketching* and it is widely used in several contexts, such as game and film industry.

In the digital entertainment industry, the most common adopted model production pipeline is the two-step development. It is accomplished manually by at least two kinds of professional artists. The first artist is a 2D illustrator that generates sketches of the scenes and for each of the objects present in it, typically from different angles - which is called *Concept Art*. Based on these drawings, another kind of professional, a 3D modeler finally generates the model for each of the objects of the scene in a laborious manual job.

Despite this development pipeline has worked for many years and the computer graphics industry is booming with many high quality objects and scenes produced, it is still very expensive as it is labor-intensive and very specialized. Our goal is to accelerate the model generation pipeline by automatically producing high quality models directly from the sketches.

Since we start from 2D images, aliasing is expected, due to discretization in the image representation. We devised techniques to alleviate the problem by smoothing out the generated surfaces. We also propose a mesh simplification algorithm to reduce the size of the generated model by reducing the number of faces on the final output. By proposing a smoothing method and a mesh simplification technique, in this paper we extended our previous work proposed in [Viana et al. 2013].

Our methodology is able to quickly generate reasonable models for several objects based upon just a few sketches. When comparing the generated models to hand-made models of the same objects, our experiments have shown that our error is less than 4.6%.

Related Works Extracting three-dimensional features from images for creating models has been a strategy widely explored in recent years [Grimm and Joshi 2012; Olsen et al. 2011; Yang et al. 2005; Yang et al. 2013]. In [Igarashi et al. 1999] it is presented a sketched based application which generates 3D models based on 2D shapes created by the user. Through a Graphical User Interface, the user draws the silhouette of objects and interactively create its 3D model. Although simple and most of the time intuitive, the Teddy application restricts the user to its set of modeling operations, which limits the variety of shapes that can be created.

Another approach is proposed in [Chen et al. 2013]. The authors use a semi-automatic process to generate 3D models based on the curvature and object composition defined by the users. Their algorithm detects the edge and uses the topological information of the object to estimate an approximation of three-dimensional data. Although this method can provide impressive three-dimensional models, it needs manual intervention. The user needs to manually define the topology and provide the details of the objects as not all are evident from the single available view of the object.

Despite the several approaches in the literature for generating 3D models from two-dimensional visualizations, we feel that none of them are adequately suited to be applied to the current manual process employed in the game industry. The digital entertainment industry is not receptive to big changes in the development process, therefore, solutions that change too much the process is hard to be implemented. Thus, the major advantage of our methodology is on practical appliance since it does not change the process used by industry, only simplifies and improves it, resulting in increased productivity and the consequent cost reduction for game development projects.

2 Methodology

This section describes our methodology for generating 3D models from the concept art of objects. For each object, a concept art is provided to the program in the form of 2D image files while the 3D model consists of a triangular mesh describing the topology (or the boundary) of the object. We execute the conversion in two phases.

These two phases are divided into 6 processing steps, as shown in Figure 1. The first 4 steps constitute phase 1, which produces the first version of the 3D model, and the final 2 steps constitute phase 2, which improves the generated model. The following subsections gives the details of each the steps.

Coarse Modeling For generating 3D models automatically from concept art, we use an approach based on Visual Hull algorithm [Laurentine 1994], which uses two or more different views of the same object to reconstruct the object in a three-dimensional space of *Voxels*.

We start with a volume divided into voxels of predefined size. For each input image, the algorithm traverses the 3D grid of voxels mapping each one to the corresponding pixel on the image. The voxel is marked as full or empty, depending on whether it sees the object on the corresponding pixel of the image or not. After processing all images, the grid of voxels will contain the first approximation of the object. The advantages of this approach is that the runtime and quality can be controlled by changing the size of the grid, and the implementation is simple.

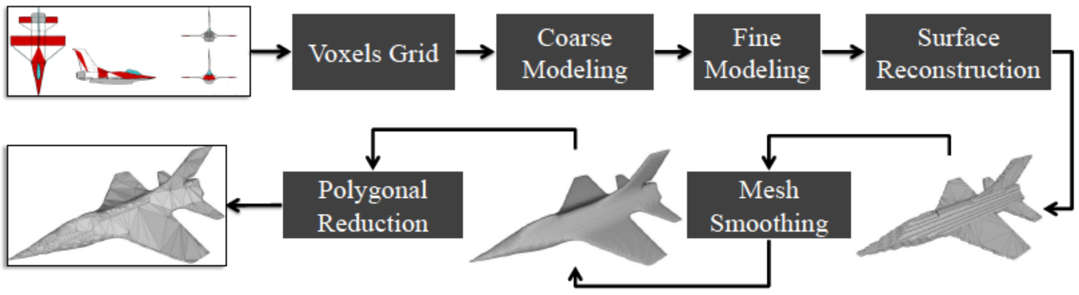

Figure 1: *Diagram with the main steps of the methodology. After determining the visibility and estimating the position of each voxel, our approach creates a triangle mesh based on the centroids of the voxels and their neighborhood smooths the mesh generated and reduces the number of polygons required to represent the 3D model.*

Fine Modeling The second step of our methodology consists of an improvement on the grid representation. This improvement is achieved by using an extension of the *Visual Hull* algorithm called *Marching Intersects* [Cignoni et al. 2001]. The idea is to traverse the border in the picture finding the intersections with the voxel surfaces. Unlike the original algorithm, which uses the voxel corner, the Marching Intersects algorithm creates a vertex in the exact location of the edge. It updates the corners of the voxel to the position of the intersections. This approach generates a better mesh by being more accurate about the contours of the object.

We propose a variation of *Marching Intersects* which is based on the strategy of going over all image and checking every voxels, verifying which ones are visible in the scene. This step is $O(CN^3)$, for a number of camera equals to C . Since, the Marching Intersects algorithm checks in all voxels what vertices belong to the mesh surface and updates their positions, spending $O(N^3) + O(v)$, where v is the number of updated vertices, the final complexity for this step is $O(N^3) + O(v)$.

Surface Reconstruction The last step of creating the 3D model is to convert the current volume representation into a three-dimensional mesh. We used the Marching Cubes algorithm [Lorensen and E. 1987] to perform this conversion. By checking which vertex belongs to a surface of the final mesh, triangles are created according to the options in a table with 256 different options for creating polygons.

Mesh smoothing To eliminate the jaggedness of the mesh, we propose an algorithm that iteratively repositions the vertices in relation to their neighbors. Our approach is based on the *Laplacian* operator defined by:

$$\mathcal{L}(V_i) = \sum_{j=1}^{N}(V_j - V_i), \tag{1}$$

where V_i is the vertex which should be smoothed, V_j is adjacent vertex.

By making use of the concept of relaxation, which is similar to traditional shortest path algorithms, we change the position of a vertex relative to the edges of neighbors vertices, to obtain a relaxed or smooth vertex position. The smoothing is performed by averaging of all adjacent vertex vectors:

$$V_i' = V_i + \frac{1}{N}\sum_{j=1}^{N}(V_j - V_i). \tag{2}$$

As the smoothing loop goes over all vertices considering their neighbors, our smooth algorithm presents $O(v + E^2)$ of complexity, where v is the number of vertices and E is the number of edges. A pseudo-code for this procedure is shown in Algorithm 1.

Polygon Reduction As aforementioned, the final step is to perform the simplification of the mesh due to the huge number of triangles generated by the Marching Cubes algorithm. This is necessary, because the processing cost for rendering a model is proportional to the number of polygons.

```
for each vertex that belongs to the mesh do
    for each adjacent vertex do
        Compute the direction to adjacent vertex;
        Sum all vectors;
    end
    Compute the average vector;
    Update the position of the current vertex;
end
```

Algorithm 1: Pseudo-code of the 3D mesh smoothing algorithm

We developed an algorithm inspired on [Zhu et al. 2005], which replaces small triangles by larger ones where the overall surface is mostly planar. To accomplish that, we included a parameter that determines the maximum curvature between adjacent triangles that can be removed. Such parameter asserts that the simplification performed in the mesh does not destroy real borders in the 3D model. Thus, our algorithm simplifies the mesh in the regions where it is not needed too much details and it is capable of maintain details when needed.

The algorithm is iterative and runs while there are triangles that can be excluded from the mesh. During the execution the algorithm uses a maximum angle between the adjacent triangles to select vertices as candidates to be removed. For each vertex candidate, the algorithm computes its importance according to the formula:

$$I(V_i) = \arccos(\langle N_a, N_b \rangle) \sum_{j=1}^{N} \| V_j - V_i \|_2, \tag{3}$$

where $I(V_i)$ is the importance of the vertex i, N_a and N_b are normals of adjacent triangles of vertex V_i and $\langle . \rangle$ is the dot product.

The vertices with low importance are removed and the importance of adjacent vertices are recomputed. This process is repeated until no vertices can be removed. To compute the importance of each vertex, the curvature of adjacent triangles are required, which makes the algorithm traverse all the edges from the adjacent triangles twice, and this is done for all vertices. Assuming that it is possible to remove the entire mesh then the cost of this step has $O(v^2 + v^2E^2)$), where v is the number of vertices and E of edges leading to the adjacent vertex.

3 Experiments

To evaluate and analyze the capability of our methodology to create high quality three-dimensional models, we performed several experiments by using concept arts of six different objects (Figure 2). We also tested our methodology with synthetic concept arts, which were created from two 3D models (Figure 3).

Smooth 3D Models In the first experiments we evaluate the quality of 3D models created by our technique considering our smoothing algorithm. Figure 5 shows the results for the non-smoothing and smoothing models. One can readily see that the

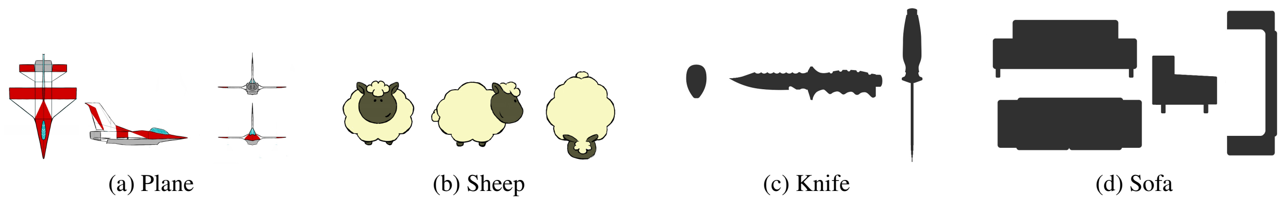

Figure 2: *Concept arts of four different objects: an airplane, a sheep, a knife and a sofa. These figures were used as input for our methodology in the experiments.*

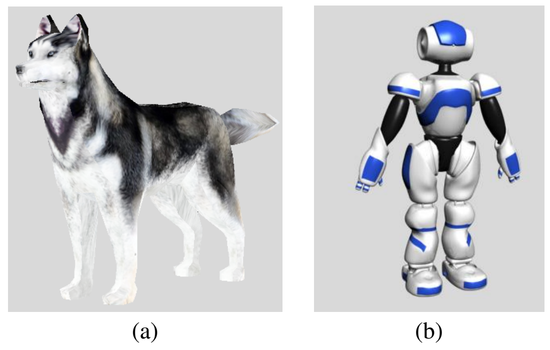

Figure 3: *3D models used validate our methodology: (a) a dog and (b) robot model.*

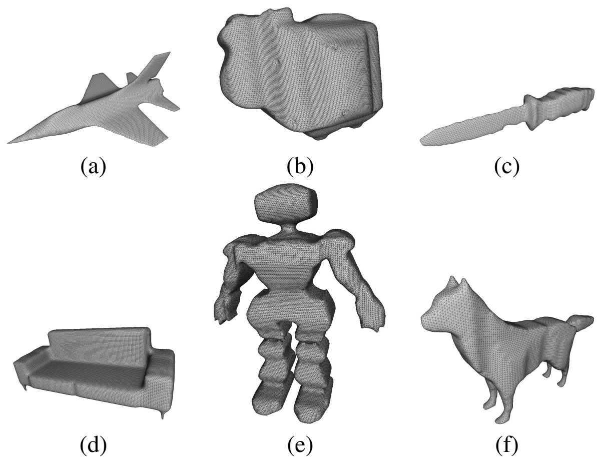

Figure 4: *Three-dimensional models created by using 2D sketches pictures.*

contours of the airplane are closer to those in the conceptual art. The smoothing method did not remove fine details and sharp features present in the airplane drawing.

Polygonal Reduction In the second test, we analyze the 3D meshes after applying our polygonal reduction algorithm. Figure 6 shows the results for the airplane. We varied the limiting angle for polygon simplification, starting with 5 degree angle and finishing with 40 degrees. The larger the angle, the more aggressive the simplification is.

By visual inspection of resulting model showed in Figure 6, we see that our algorithm is quite good at avoiding breaking the overall contours of the models, although some deformation happens for more aggressive simplification. Intuitively, our approach does only local analysis on the triangles and their neighbors. Somewhat more global analysis should yield better results by being more aggressive while preserving important features of the objects.

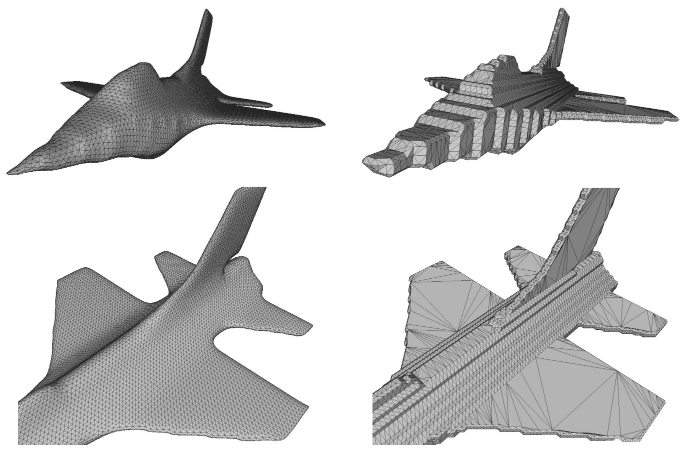

Figure 5: *Airplane model before and after applying smoothing algorithm.*

Execution time On average, modelers spend up to a day to come up with models for examples like Figure 2. With our methodology, the creation of models for airplane, knife, sofa and sheep sketches spends 39, 12, 196 and 243 seconds respectively (Intel Core i5, 4GB).

Validation At last, in order to validate the quality of the generated models, we executed an experiment by using a couple of handmade models (Figure 3): a dog model with $1,544$ and the humanoid robot with $17,906$ polygons respectively. We generated images from these two models from three different viewpoints and used them as input for our methodology. Our aim is to verify whether our approach is capable of generating models with similar quality as the hand-made ones. The results for the robot and dog models are presented in Figures 4 (e) and (f).

Quantitative analysis We further generated image from the views of the newly created models and compared then against to the images produced from views of the original models. We used different resolutions of the grid of voxels, ranging from 8 times smaller than the image resolution to 4 times smaller. A grid with high resolution makes the whole process more capable of extracting details from the views, however, also increase the processing time.

The Figure 7 shows the error in respect to the resolution of voxel grid. As expected, the quality of models improves as the resolution of the grid is closer to the image resolution. The algorithm presented on average 20.78 % of errors for the voxel grid with 8 times less resolution and 3.78 % of errors for voxel grid with a resolution 4 times lower resolution than the original images.

As aforementioned, another important property for our methodology is the processing time. Figure 7 shows the time spent by the our method generate 3D models. The time is plotted according to the size of the voxel grid. From the plot, it is possible to note that even for a large grid (4 times smaller than the image) the processing time is acceptable.

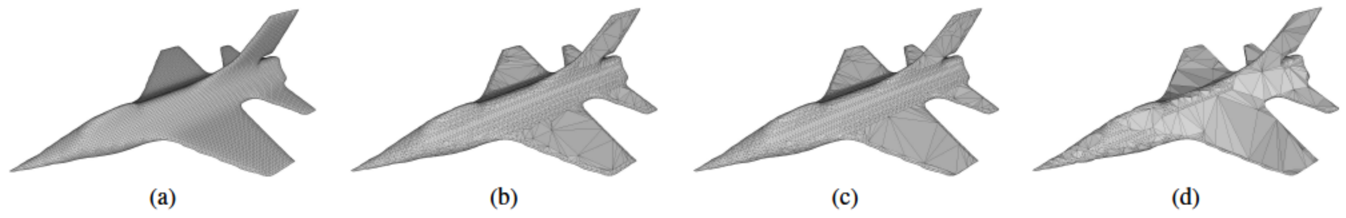

Figure 6: *Airplane 3D models resulting from the polygonal reduction. The rightmost image is the model for high density of triangles and the leftmost is the model after applying the polygonal reduction method. The model shape is preserved even for dramatic decreases in the number of polygon.*

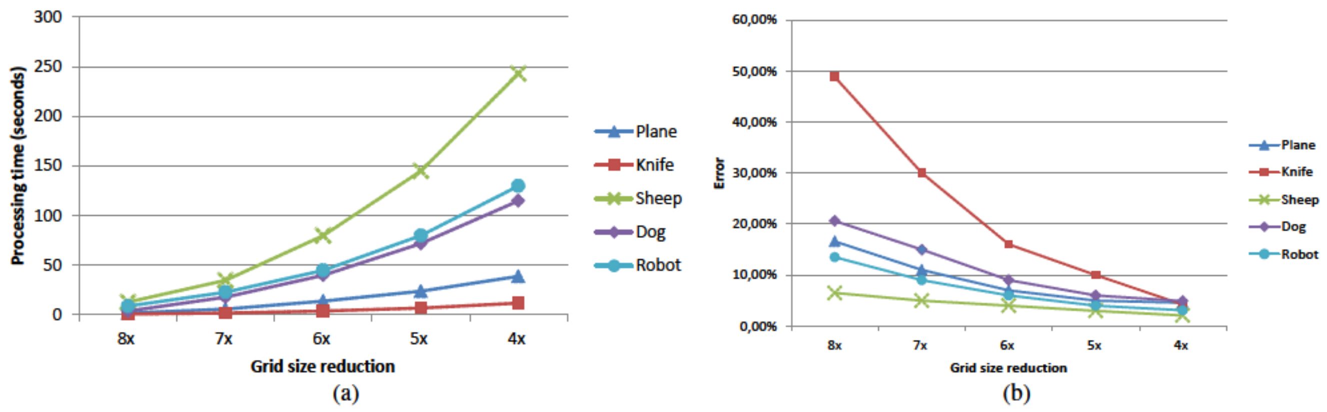

Figure 7: *(a) The figure shows the plots for the processing time and for (b) the errors of 3D models in relation to the size of the voxel grid.*

4 Conclusion

This paper presents a methodology that can be used to automatically generate 3D models directly from concept art. We presented modifications to the *Marching Intersections* algorithm that improves the approximation of the volume information from 2D images and a technique capable of smoothing the 3D mesh which removes sharp and alias areas from 3D mesh. Also, we proposed a new polygon reduction algorithm in order to remove unnecessary polygons. The experiments showed that our methodology is able to generate models with small amount of errors only using simple pictures. On average the models created by our method presented errors from 2% to 5%. However, by increasing the resolution of the grid these errors shows a downward trend. As far as time is concerned the algorithm was able to generate 3D models from 2D views of an object at a time that it can improve the manual process currently used in the industry.

As future work we plan to extract the textures from images and apply them in the generated 3D model. Also, we intend to use the modern solution Dual Marching Cubes [Schaefer and Warren 2004], which produces more accurate results in the extraction of polygons from the structure of voxels.

Acknowledgements

The authors would like to thank CAPES, FAPEMIG and CNPq for the financial support provided.

References

CHEN, T., ZHU1, Z., SHAMIR, A., HU1, S.-M., AND COHEN-OR, D. 2013. Sweep: Extracting Editable Objects from a Single Photo. *SIGGRAPH ASIA*.

CIGNONI, P., GANOVELLI, F., MONTANI, C., PINGI, P., AND SCOPIGNO, R. 2001. Marching Intersections: an Efficient Resampling Algorithm for Surface Management. *Shape Modeling and Applications, SMI 2001 International Conference on.*, 296–305.

GRIMM, C., AND JOSHI, P. 2012. An Interface for Sketching 3D Curves. *EUROGRAPHICS Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling 1*, 121–130.

IGARASHI, T., MATSUOKA, S., AND TANAKA, H. 1999. Teddy: A Sketching Interface for 3D Freeform Design. *ACM SIGGRAPH 1*, 1–8.

LAURENTINE, A. 1994. The Visual Hull Concept for Silhouette-Based Image Understanding. *Transaction on Pattern Analysis and Machine Intelligence 16*, 2.

LORENSEN, W. E., AND E., H. 1987. Marching Cubes: A high resolution 3D surface construction algorithm. *In: Computer Graphics 21*, 4.

OLSEN, L., SAMAVATI, F., AND JORGE, J. 2011. Naturasketch: Modeling from images and natural sketches. *IEEE Computing Graphics Appl. 31 1*, 24–34.

SCHAEFER, S., AND WARREN, J. 2004. Dual Marching Cubes: Primal Contouring of Dual Grids. *Proceedings of the Computer Graphics and Applications, 12th Pacific*, 70–76.

VIANA, R. D., FERREIRA, R. A. C., AND NASCIMENTO, E. R. 2013. Utilização de arte conceitual para geração de modelos 3D. In *Proceedings of the XII Brazilian Symposium on Games and Digital Entertainment - SBGames 2013*.

YANG, C., SHARON, D., AND VAN DE PANNE, M. 2005. Sketch-based Modeling of Parameterized Objects. In *ACM SIGGRAPH 2005 Sketches*, ACM, New York, NY, USA, SIGGRAPH '05.

YANG, L., LIU, J., AND TANG, X. 2013. Complex 3D General Object Reconstruction from Line Drawings. In *ICCV*, 1433–1440.

ZHU, J., TANAKA, T., AND SAITO, Y. 2005. A New Mesh Simplification Algorithm for the Application of Surface Sculpture.