Non-Photorealistic Rendering in Customizable Styles for Mobile Collaboration

Rong-Qin Chen[1, 2], Min Tang[1], Jin-Xiang Dong[1]
[1]*College of Computer Science and Technology, Zhejiang University, Hangzhou, P.R. China, 310027*
[2]*School of Mathematics and Information Engineering, Taizhou University, Linhai, P.R. China, 317000*
chen_rongqin@yahoo.com.cn, tang_m@zju.edu.cn, djx@zju.edu.cn

Abstract

Mobile devices quickly become popular tools in collaborative design for their portability and supporting of graphics user interface. However, their limited resources may result in the complexity of rendering 3D products. Non-photorealistic rendering is extremely suitable for mobile devices since they efficiently save computational power and convey visual information. In this paper, a new approach is proposed to deal with mobile collaboration based on non-photorealistic rendering. In contrast to traditional methods, our method describes the user's design intent in an XML-based model, which is customizable and improves the collaboration between mobile users.

Keywords: NPR, Mobile Collaboration, Customization, XML.

1. Introduction

With the popularization of networks and globalization, the manufacturing industries are facing fierce competition and needing more collaboration to accelerate the product development. Meanwhile, the pervasiveness of mobile devices such as Pocket PCs, Personal Digital Assistants (PDAs) and cellular phones has increased steadily in recent years. Mobile collaborative design may accelerate the product development because of the portability of mobile devices. Generally speaking, the collaboration should be made by rendering product models on mobile devices in virtue of the power of some 3D graphics APIs. OpenGL ES is now the standard for embedded accelerated graphics and OpenGL ES 2.0 enables full programmable 3D graphics [1]. However, rendering 3D product models on mobile devices is still a formidable task because of the following reasons: 1) The resources on mobile devices are extremely scarce: limited bandwidth, small amounts of memory, weak processing power, low resolution display and so on. 2) The power of rendering 3D models with OpenGL ES generally relies on hardware acceleration that is not supported on the mainstream mobile devices.

The strategies to solve the resource limitation of mobile devices generally refer to the model simplification or image-based rendering [2,3]. In our method, only the 2D wire-frame model [4] is transferred and rendered with 2D graphics API so that it is suitable to most mobile devices.

Moreover, in the early stages of the product design process, design sketches are often drawn quickly in order to visualize the rough geometric shapes of prospective products. Traditional collaborative CAD systems are often too tedious to allow the quick expression of design idea, thus interfering with the collaboration. Non-photorealistic rendering [5,6,7] is better than photorealistic rendering in conveying visual information and design intent because it can provide hand-drawn style that preserves the user's design idea.

The rest of this paper is organized as follows. Section 2 describes the previous work related to the graphics techniques on mobile devices. The overview of the proposed approach is described in Section 3. Section 4 presents the detailed procedures of XML-based representation. In section 5, the non-photorealistic rendering algorithms are explored. Section 6 provides details on our implementation. Section 7 reports results of our method and finally Section 8 concludes and describes opportunities for future work.

2. Related work

2.1. Rendering methods on mobile devices

One of the important issues in product design is how to render the models with graphics APIs on mobile devices. Many rendering methods that implemented on desktop personal computers now turn to mobile devices [2, 3,5,6,7,8]. Some of them use non-photorealistic rendering techniques to achieve visually appealing results at interactive rates [5,6,7].

Daniel Cohen-Or, et al [8] developed a server-based remote walkthrough system, the server holds the large environment, generates the frames, encodes and transmits them to the client. The encoded frames are transmitted as video streams (MPEG-4) to the client, then decoded and displayed in the client. The layering techniques are used to yield a lighter stream, which can minimize the transmission delay due to the narrow bandwidth. However, the client must be provided with a MPEG-4 chip, which reduces the adaptability of this system.

Considering that the display of a mobile device is smaller than that of a personal computer and the runtime of image-based rendering method mainly depends on the display resolution, [2,3] proposed an image-based rendering method, which is independent on geometric complexity and capable of rendering complex 3D models. However, the image-based rendering method increases the workload of the system and is not suitable for simple models.

Roman Zenka, et al [5] proposed a method that allows users' walkthroughs in 3D scenes with a Macromedia flash player. It renders a 3D scene into vector images on the server and transfers these data to the client as a movie. Moreover, non-photorealistic rendering techniques are used for more eye-pleasing results and higher data compression.

A fatal disadvantage of the above-mentioned methods is that the transmitted data are non-geometrical, so the users cannot control the models on mobile devices flexibly. D. Hekmatzada, et al [6] developed a server-client system. An important feature of their system is the use of progressive level of detail (LOD) representations for the transmitted models, which enables coarse renderings of the models to be quickly loaded even at the limited bandwidth. Because the transmitted models are polygonal, users can manipulate the models arbitrarily. However, the transmitted data contain too many vertexes and faces for complex models, so the client must wait if the user wants to view and manipulate a detailed portion. To overcome this shortcoming, the system also provides the option to render the visible silhouette edge segments relying on the client's position and view direction in each frame. To reduce the data size of the models, Joachim Diepstraten, et al [7] proposed a method to derive 2D lines from feature lines of 3D objects and render them on mobile devices. But it is helpless for collaborations for the absence of NPR style information in the 2D lines.

2.2. A customizable rendering method on mobile devices

In mobile collaboration, the rendering method to be used for product models relies on characteristics of models, capabilities of mobile devices, intent of designers and so on. A customizable rendering method can satisfy these different requirements.

Extensible Markup Language (XML) [9] plays an important role in information exchange. XML-based representation can effectively describe structured or semi-structured data and convert from other kinds of representations with many available tools, which is more helpful for manipulating, sharing and reusing the model data among different systems than other representations.

In this paper, an XML-based server-client system is developed to achieve the purpose of customization according to the user's design intent. Moreover, some non-photorealistic rendering algorithms are proposed and implemented, which allow being flexibly controlled and combined by the users.

3. An overview of the proposed approach

3.1. A framework of XML-based server-client system

Here we present a framework of XML-based server-client system quite suitable to the early stages of design. The three-dimensional models are stored at the server side application. Without loss of generality, two mobile users "A" and "B" (client side applications) are cooperatively designing one product. When "A" makes an operation on the product, it encapsulates the operation and a NPR style to deliver the design intent in XML format, and then sends the XML action to the server side. Once the server side application receives the XML action from "A", it first parses it with an XML parser, performs corresponding operation on current 3D model, abstracts 2D Lines from the resultant 3D model and feedbacks the XML-based representation for NPR styles model to "B". "B" then parses the XML model and renders it with non-photorealistic rendering algorithms. Finally, "B" will catch the design intent of "A" since NPR methods have the advantage in conveying visual information. The overall process of our system framework is shown in Figure 1.

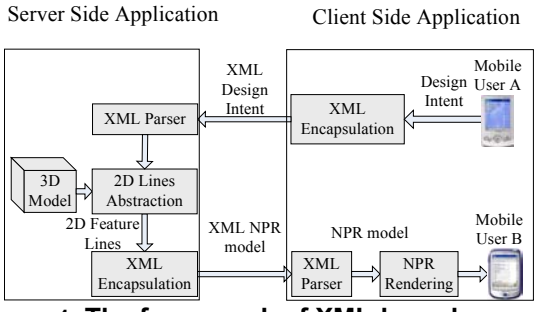


Figure 1. The framework of XML-based server-client system

3.2. Strategies to implement the system framework

There are two strategies to implement the above-mentioned framework:

1) The client side applications communicate with the server side application through the "SOCKET" protocol. This strategy can be applied in Local Area Network (LAN).

2) The client side applications are special web browsers that support XML grammar and use the "HTTP" protocol to communicate with the server side application. This strategy can escape from the "Firewall" and can be applied in Wide Area Network (WAN).

4. XML-based representation for 2D models

Generally, there are some approaches widely used to represent 3D models, such as boundary representation, facet-based representation, wire-frame based representation and feature-based representation [4]. Among these approaches, the wire-frame based representation appears the most preferable one to mobile devices because it only contains some feature lines, such as silhouettes, creases and border edges.

To minimize the transferred data, our system adopts wire-frame based representation models and projects them to 2D window coordinate system. A 2D model contains some 2D lines, each line contains two vertexes and each vertex contains two coordinates. In addition, some lines linearized from a curve are merged into polygonal lines, so a line can contain lots of vertexes in the transferred model of our system. The 2D model can be represented as a tree shown in Figure 2.

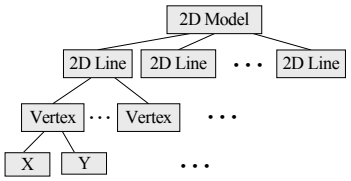


Figure 2. The hierarchy of 2D tree model

4.1. XML representation of model data

XML can represent structured information due to its comprehensibility and extensibility, thus the above-mentioned 2D tree model and its hierarchy can be described with XML texts. In our system, some XML elements corresponding to the nodes in the tree model are defined. The XML expression of the hierarchy of 2D tree model is described as follows:

```
<MODEL>
<LINE>
<POINT>
<X>X coordinate value</X>
<Y> Y coordinate value </Y>
</POINT>
…
</LINE>
…
</MODEL>
```

With XML-based representation, incremental transmission [9] can be easily integrated into some collaborative systems. When performing an operation on the current model, the client just updates the revised parts from the server in XML format, merges into the original XML texts and renders the resultant XML model.

4.2. XML representation of NPR styles

NPR styles will be continuously created by great human imaginations. In this paper, some line styles including "zigzag style", "wavy style", "quick style" and "rough style", width styles and thickness styles are proposed. To describe NPR styles in XML, the following work will be done:

1) Define some attributes: "*linestyle*" is used to describe line styles; "*startwidth*", "*midwidth*", "*endwidth*" and "*widthscale*" are used to describe width styles; "*startthick*", "*midthick*", "*endthick*" and "*thickscale*" are used to describe thickness styles.

2) Add some restrictions to the defined attributes: "*linestyle*" can be "zigzag", "wavy", "quick" or "rough"; "*startwidth*", "*midwidth*" and "*endwidth*" are positive integers; "*startthick*", "*midthick*", "*endthick*", "*widthscale*" and "*thickscale*" are real numbers ranging from 0.0 to 1.0.

3) The algorithm of each style is implemented in the client side application. If a new style is created, the algorithm should be implemented.

The NPR styles of a line or a model are evaluated in parsing an XML model. Our system allows the different styles between one node and its brother nodes, so a model can be rendered flexibly according to the design intent. Figure 3 shows different styles of lines in a cube model.

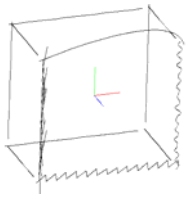


Figure 3. Different line styles in a model

4.3. XML representation of multi-styles

It is important to render a model in multi-styles in order to effectively convey design intent or achieve perfect visual effect. Our system allows representing multi-styles in an XML node (a line node or a model node). The different combinations of styles with adjustable parameters lead to a large quantity of

appearances of the same model. The following XML texts present a model and one of lines being described in multi-styles.

<MODEL linestyle="zigzag" startthick="1.0" Midthick="0.0" endthick="1.0" thickscale="0.5">
<LINE lineStyle="wavy" startwidth="5" midwidth="1" endwidth="5" widthscale="0.5">
…
</LINE>
…
</MODEL>

5. Non-photorealistic styles based on 2D lines

5.1. Line styles of 2D models

In our system, several line styles are created including zigzag style line, wavy style line, quick style line and rough style line.

1) Zigzag style line: a line that proceeds by sharp turns in alternating directions and shows the power of strength. To draw a line in zigzag style, we split the straight line into some segments and each segment is replaced with two joint lines. Figure 4 shows that a segment "CD" is replaced with "CE" and "ED".

2) Wavy style line: a line that proceeds by smooth turns in alternating directions and shows the elegancy and streamline shape of a dynamic curve. We use two Bezier curves to replace the segment. In Figure 5, the segment "CE" is replaced with two Bezier curves which controlled by two polygons ("p1p3p5" and "p2p4p5").

3) Quick style line: a line that drawn by a quickly moving pen shows the springiness. We use a Bezier curve (such as the Bezier curve controlled by the polygon "ADB" in Figure 6) to replace the whole straight line. If "$\angle$ABD" is an obtuse angle, the quick style line appears a hook in the end, which often owned by hand-drawn lines [10] (shown in Figure 7).

4) Rough style line: a line that hastily drawn as a preliminary study for sketch. We use some cross lines to replace the segments of a straight line.

As to the segments of polygonal lines, the zigzag, wavy or rough style is similar to the segments of straight lines while quick style is dealt discriminatingly. When rendering polygonal lines in quick style, the algorithm computes the obliquity $\theta_i, i = 1,2,\cdots n$ of each segment $S_i, i = 1,2,\cdots n$, calculates the difference of obliquity $\Delta\theta_i = \theta_{i+1} - \theta_i, i = 1,2,\cdots n-1$ and cumulates these differences iteratively:

$$\begin{cases} A_1 = \Delta\theta_1 \\ A_{i+1} = A_i + \Delta\theta_{i+1} \end{cases}$$

Given the threshold $T>0$, if the condition $A_i < T$ is satisfied in the process of iteration, the segments $S_1, S_2, \cdots, S_i$ of polygonal lines are replaced with a Bezier curve (controlled by the polygon "p1p3p2" shown in Figure 8)

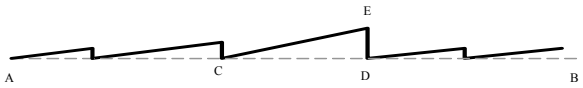


Figure 4. A straight line in zigzag style

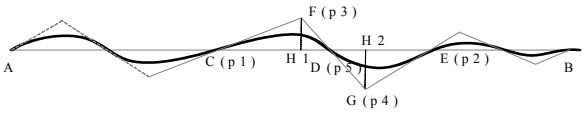


Figure 5. A straight line in wavy style

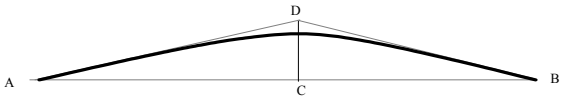


Figure 6. A straight line in quick style

Figure 7. A quick style line with a hook in the end

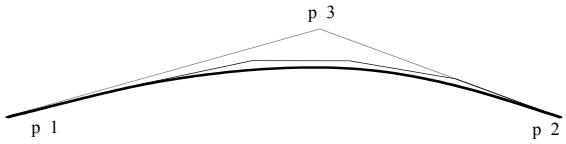


Figure 8. The polygonal lines

5.2. A multi-style algorithm

XML representation allows describing multi-styles, so the multi-style algorithm must be implemented in the client. In our current system, the width and thickness styles are combined into the line styles. The algorithm is described as follows:

1) Split the straight line into several segments. (Ignore this step for polygonal lines)

2) Calculate the width or thickness value of each segmentation point if the width or thickness style is taken into account.

3) Draw all segments of the line in the given line style and change the width or thickness value before drawing the next segment.

The proposed algorithm is not suitable for the quick line style, so the width or thickness style is combined with the quick line style in drawing segments of the Bezier curve. If a new style (such as a color style) is

introduced, it also can be taken into account in this algorithm.

6. Implementation details

The implemented XML-based server-client system consists of a desktop personal computer (server) and a PDA develop board (client). The configuration of the server is 742 Megahertz CPU, 256 Megabytes RAM, display card with 16MB of VRAM, 100Mbit switched Ethernet and Windows XP operating system. The PDA develop board is equipped with 32bit Intel XScale PXA255 Processor, 64MB SDRAM, 32MB Flash Memory, 10 Base-T Ethernet, TFT 6.4" Color LCD and Embedded Linux operating system.

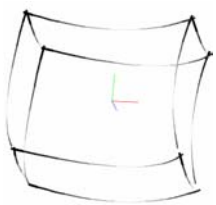
The WinSock API in the server side application and Linux socket API in the client side application to communicate with each other based on the TCP/IP protocol. The server side application needs the ACIS kernel to perform the modeling and abstracting the feature lines from the models, OpenGL to display the models and obtain the 2D data in feedback rendering mode. Moreover, the client side application needs the Xlib that is usually supported by the Linux operating system.

7. Results

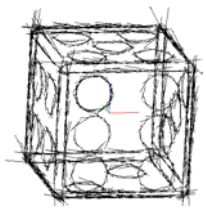
Figure 9 presents some models being rendered in different non-photorealistic styles on our mobile device. These styles are customizable in our system because users can flexibly control the parameters of each style and combine different styles with each other.

Table 1 presents the comparison about the data sizes of the facet-based and wire-frame based models. We found that the data size of the wire-frame based models is less than that of the facet-based models, thus the wire-frame based representation is more suitable for mobile devices.

Moreover, the comparison about the efficiency of each line styles is presented in Figure10 according to the response time when one client interacts with the other client. The response time consists of three main parts: the time of sending actions to the server, the time of server's processing and transmitting the resultant model to a client, the time of client's rendering model. The comparison indicates that our methods are fairly satisfied with the requirements of interaction.



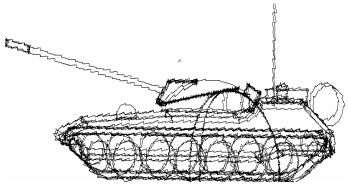
(A) A cube in quick, width and thickness styles



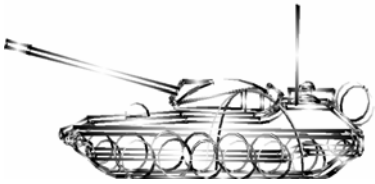
(B) A dice in rough styles

(C) A bat in wavy, width and thickness styles

(D) A bat in rough, width and thickness styles



(E) A tank in zigzag style



(F) A tank in width and thickness styles

Figure 9. Some models being rendered in different non-photorealistic styles

Table 1. Comparison about the data size of facet-based and wire-frame based models

Data size \ Models		Cube	Bat	Dice	Tank
Facet-based models	Lines	36	5712	15474	27690
	Vertexes	72	11424	30948	55380
	XML file	8k	948k	2560k	4852k
Wire-frame based models	Lines	12	11	95	699
	Vertexes	24	111	412	2873
	XML file	4k	12k	32k	228k

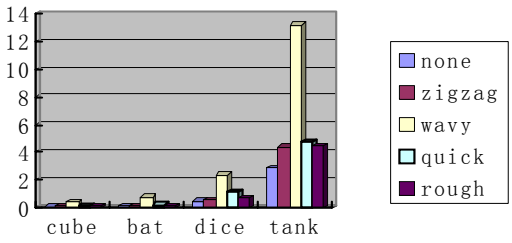


Figure 10. Comparison about the response time of different line styles

8. Conclusions and Future work

Our main contributions are:

1) The XML-based representation for NPR styles model can conveniently express design intent and improve the collaboration between two mobile users.

Also, The XML models improve the comprehensibility and extensibility of our system and can be well integrated into other systems.

2) The combination of different styles in an XML model brings the advantage of conveying sufficient visual information in our system

3) Several non-photorealistic rendering algorithms are proposed and implemented.

However, our current system also has some limitations, the obvious one is that it is dependent on the geometric complexity. To support real-time interaction, our system just renders the bounding box of the model in the course of interaction. Our future work will focus on the following aspects:

1) In the current implementation, the performance is not our main objective and is not completely optimized. The performance of the server side application can be improved by hardware acceleration and the rendering algorithms in the client should be further optimized.

2) In the current implementation, hidden-line removal is not taken into account. Certain hidden-line removal algorithm can reduce the data size of the transmitted XML model and cut down the transmission delay of our system, so it should be implemented urgently.

3) To better convey the mobile users' design intent, more suitable NPR styles should be added to our system.

Acknowledgement

The project was supported in part by National Key Basic Research and Development Program (2006CB303106), Science & Technology Bureau of Taizhou City in Zhejiang Province (063ky08).

References

[1] Kari Pulli Nokia Research Center & MIT, Jani Vaarala Nokia, Ville Miettinen Hybrid Graphics, et al. Developing Mobile 3D Applications with OpenGL ES and M3G[C]. SIGGRAPH 2005.

[2] C. Chang, S. Ger, Enhancing 3D Graphics on Mobile Devices by Image-Based Rendering, *Proc. IEEE Pacific-Rim Conf. on Multimedia,* 2002, pages 1105-1111.

[3] Yu Lei, Zhongding Jiang, Deren Chen, Hujun Bao, Image-Based Walkthrough over Internet on Mobile Devices, *GCC Workshops 2004*, pages 728-735.

[4] Youngjae Song and Kunwoo Lee, Incremental Transmission of B-Rep Models through the Network, *Computer-Aided Design and Applications, Vo.1, No.1-4, Proc. of CAD'04 (the 2004 International CAD Conference and Exhibition)*, May 24-28 2004, Pattaya, Thailand, pages 523-529.

[5] Roman Zenka, Pavel Slavík, Non-Photorealistic Walkthroughs Using Flash, *WSCG (Posters) 2004*, pages 201-204.

[6] D. Hekmatzada, J. Meseth and R. Klein, Non-Photorealistic Rendering of Complex 3D Models on Mobile Devices, *In 8th Annual Conference of the International Association for Mathematical Geology*, volume 2, Alfred-Wegener-Stiftung, September 2002, pages 93-98.

[7] J. Diepstraten, M. Gorke and T. Ertl, Remote Line Rendering for Mobile Devices, *In Proceedings of IEEE Computer Graphics International (CGI)'04,* 2004, pages 454-461.

[8] D. Cohen-Or, Y. Noimark, and T. Zvi, A Server-based Interactive Remote Walkthrough, *In Proc. of the 6th Eurographics Workshop on Multimedia*, 2001,pages 75–86.

[9] A. Deutsch, M. Fernandex, D. Florescu, A.Levy and D. Suciu, A Query Language for XML, *In International World Wide Web Conference*, 1999.

[10] Thomas Strothotte, and Stefan Schlechtweg, *Non-Photorealistic Computer Graphics: Modeling, Rendering, and Animation*, Elsevier Science, 2004.