The design and implementation of stereoscopic 3D scalable vector graphics based on WebKit

Zhongxin Liu, Wenmin Wang*, Ronggang Wang
School of Electronic and Computer Engineering, Shenzhen Graduate School, Peking University, Lishui Road 2199, Nanshan District, Shenzhen, 518055 China

ABSTRACT

Scalable Vector Graphics (SVG), which is a language designed based on eXtensible Markup Language (XML), is used to describe basic shapes embedded in webpages, such as circles and rectangles. However, it can only depict 2D shapes. As a consequence, web pages using classical SVG can only display 2D shapes on a screen. With the increasing development of stereoscopic 3D (S3D) technology, binocular 3D devices have been widely used. Under this circumstance, we intend to extend the widely used web rendering engine WebKit to support the description and display of S3D webpages. Therefore, the extension of SVG is of necessity. In this paper, we will describe how to design and implement SVG shapes with stereoscopic 3D mode. Two attributes representing the depth and thickness are added to support S3D shapes. The elimination of hidden lines and hidden surfaces, which is an important process in this project, is described as well. The modification of WebKit is also discussed, which is made to support the generation of both left view and right view at the same time. As is shown in the result, in contrast to the 2D shapes generated by the Google Chrome web browser, the shapes got from our modified browser are in S3D mode. With the feeling of depth and thickness, the shapes seem to be real 3D objects away from the screen, rather than simple curves and lines as before.

Keywords: Scalable Vector Graphics (SVG), stereoscopic 3D (S3D), WebKit

1. INTRODUCTION

Scalable Vector Graphics (SVG), as a standard developed by World Wide Web Consortium (W3C), is a widely used language to describe 2D graphics. It is a web development language based on eXtensible Markup Language (XML). SVG enables users to describe animated images as well as static vector graphics, with high-quality graphics from real time data [1] [2].

SVG provides users a method to describe graphics conveniently, such as circles, rectangles, polygons, ellipses, etc. Users can also put some circles, lines and other shapes into a group, and give it name to represent a group of graphics. As it is convenient to provide some meaning to it, SVG has become increasingly popular and have been used on computers, mobile phones and embedded systems.

As a language designed to describe 2D graphics in a simple method, SVG specification does not have the ability to portray graphics of 3D mode. Lots of techniques have been developed to solve this problem. Much attention has been paid to create a sense of solid geometry with a projection from 3D real world onto a flat plane. In this way, 2D SVG graphics seem to be in 3D mode when they are drawn on the screen. Another method has been developed to create 3D graphics from simple 2D shapes. By inserting new objects between two original shapes, complicated 3D graphics can be got with a method of interpolation.

However, the available mechanisms discussed above are useful to create monoscopic 3D SVG objects, rather than stereoscopic 3D (S3D) SVG objects. As S3D displays are becoming widely used recently, it is of great necessity to provide a technique to display graphics vividly on these S3Ddevices. Accordingly, it is of great importance to describe S3D graphics with a simple method that is easy to use and understand.

We took a step further to support S3D mode of SVG graphics. In this paper, we proposed a method of 2D mode to S3D mode conversion for supporting stereoscopic 3D SVG graphics, with depth and thickness attributes. We also discuss the

*wangwm@ece.pku.edu.cn; phone 86 755 2603-5700; fax 86 755 2603-2100; ece.pku.edu.cn

Stereoscopic Displays and Applications XXV, edited by Andrew J. Woods, Nicolas S. Holliman, Gregg E. Favalora,
Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 9011, 90111R · © 2014 SPIE-IS&T
CCC code: 0277-786X/14/$18 · doi: 10.1117/12.2038681

elimination of hidden lines and hidden surfaces. S3D SVG graphics, rendered by our modified rendering engine, seem to be real objects, rather than simple shapes on a flat plane.

This paper is organized as follows. Firstly, we describe in Section 2 the related techniques to generate graphics of 3D mode using the language of SVG. Then in Section 3, we explain the algorithm to extend 2D SVG graphics into stereoscopic 3D mode and eliminate hidden lines and hidden surfaces. Section 4 discusses our extensions of SVG to support S3D mode. Section 5 provides a detailed discussion about our modifications in WebKit to implement the presentation of stereoscopic 3D shapes. Section 6 provides an experiment to explain the effect when a group of three SVG shapes are rendered using our modified rendering engine.

2. RELATED WORKS

Previously, much attention has been paid to present monoscopic 3D graphics in 2D mode. Lutz Tautenhahn, a German, developed a free JavaScript script library called SVG-VML-3D [3], which is able to select SVG or VML to display 3D graphics automatically, according to the browser. A lot of functionalities, such as 3D objects displaying, scaling and rotating as well as mouse event interaction can be supported by this script library.

Some extensions have been also made to support 2.5D or 3D graphics, either used on embedded platforms or WebGIS. Jun Fujisawa and Anthony Grasso extended SVG to support 2.5D effect in SVG tiny, which is mainly used for designing mobile GUI [4]. This effect is achieved by changing the values of "scale" and "translation" attributes of the SVG graphics. David Dailey extended SVG with an element <replicate> [5]. Users can produce lots of new objects to implement 3D graphics with it. Lin Wei-Yong and Li Yan improved this approach to create 3D Digital Evaluation Model (DEM) [6]. It is helpful to establish 3D DEM, even if the contours in the datasets have different number of points.

An extension to SVG, called Constraint Scalable Vector Graphics (CSVG), provides a method to describe figures flexibly [7]. Instead of being specified in absolute terms, images are able to contain objects whose positions and other properties are specified in relation to other objects using constraints.

Some other researches about SVG have been done as well. SVG has also been used as a method to help understand and analyze complex data. A general framework has been provided, so that complex heterogeneous spatial-temporal data sets can be integrated effectively and presented with web technologies [8]. Some technologies used to present 3D web are summarized and a new 3D webpage model is proposed [9].

However, as is mentioned above, these previous works only implement the presentation of monoscopic 3D SVG graphics. Our contribution is the improvement to support stereoscopic 3D SVG graphics.

3. PRINCIPLES OF THE PROJECT

3.1 2D to Stereoscopic 3D Conversion

In order to let users see objects with S3D mode, corresponding graphics need to be calculated according to the expected object. A coordinate system is established with x-axis along with the viewing screen, z-axis perpendicular to the viewing screen. Let the position of the expected point be $P\ (x, y, z)$. Let $Pl\ (x_l, y_l)$ and $Pr\ (x_r, y_r)$ be related points on the left and right views corresponding to the point P. Let e be the distance between the eyes. Let V be the distance from the eyes to the viewing screen. According to the attributes of similar triangles, the relations among the coordinates of points P, Pl and Pr can be illustrated as (1) and (2).

$$\frac{x_l + \frac{e}{2}}{x + \frac{e}{2}} = \frac{y_l}{y} = \frac{V}{V + Z} \tag{1}$$

$$\frac{x_r - \frac{e}{2}}{x - \frac{e}{2}} = \frac{y_r}{y} = \frac{V}{V + Z} \tag{2}$$

According to equations (1) and (2), the coordinate of points $Pl\ (x_l, y_l)$ and $Pr\ (x_r, y_r)$ on the left and right views can be calculated. Their values can be calculated by simplifying these two equations shown above. In this way, users may

think there is a point at the position of P, when the left and right view images are shown on a screen of S3D mode. By changing the z-coordinate of the expected points, different shapes can be displayed at different distances, with a feeling of depth.

In order to support the representation of S3D SVG graphics, it is necessary to extend 2D SVG graphics into 3D mode. We define the outside of a three dimensional figure as the front surface and the inside as the back surface. Without loss of generality, the original 2D SVG shapes are assumed to be the front surface of 3D figures. Points on the back surface are calculated according to the points on the front surface. Let d be the thickness attribute of the expected graphics. Let $PF\ (x_f, y_f, z_f)$ be the coordinate of a point on the front surface. Let $PB(\ x_b, y_b, z_b)$ be the coordinate of the corresponding point on the back surface. Let $Pv\ (x_v, y_v, z_v)$ be the coordinate of the vanishing point. With the property of similar triangles and the definition of thickness, relations among these points can be illustrated as equations (3) and (4).

$$\frac{x_b - x_f}{x_v - x_f} = \frac{y_b - y_f}{y_v - y_f} = \frac{z_b - z_f}{z_v - z_f} \tag{3}$$

$$z_b = z_f + d \tag{4}$$

According to equations (3) and (4), the coordinates of points on the back surface can be got.

3.2 Elimination of Hidden Lines and Hidden Surfaces

It is of great necessity to eliminate hidden lines and hidden surfaces. Theoretically, if a line connecting a point on a surface and an eye intersects with another surface, the former surface is considered to be a hidden one. A similar definition can be got for a hidden line. A line segment on the contour of a hidden surface can only be considered as a candidate for a hidden line. While a line segment is considered to be visible if it is a portion of a visible surface. Normal vectors of surfaces are calculated to determine the hidden surfaces. If there is no positive component of the normal vector of a surface towards the users' eyes, the surface is considered to be a hidden one. In this way, the visible part of the contour is determined, which is enough to draw the portion of graphics that can be seen.

Suppose there are n vertexes on the front surface of a graphics in 3D mode. Let the points on the front surface be F_i, $i = 0, 1, ..., n-1$, in the clockwise direction. Let the corresponding points on the back surface be B_i, $i = 0, 1, ..., n-1$, in the clockwise direction as well. Let the vector from F_i to $F_{(i+1)\%n}$ be $\boldsymbol{FV_i}$. Let the vector from F_i to B_i be $\boldsymbol{BV_i}$. Let the normal vector of the surface $F_iF_{(i+1)\%n}B_{(i+1)\%n}B_i$ be $\boldsymbol{N_i}$. Let the vector from the center of the surface $F_iF_{(i+1)\%n}B_{(i+1)\%n}B_i$ to the users' viewpoint be $\boldsymbol{D_i}$. Let the included angle of $\boldsymbol{N_i}$ and $\boldsymbol{D_i}$ be θ. Then we can get the relations depicted by (5) and (6).

$$\boldsymbol{N_i} = \boldsymbol{FV_i} \times \boldsymbol{BV_i} \tag{5}$$

$$|\boldsymbol{N_i}|\,|\boldsymbol{D_i}|\cos\theta = \boldsymbol{N_i} \bullet \boldsymbol{D_i} \tag{6}$$

If $\cos\theta > 0$, the surface $F_iF_{(i+1)\%n}B_{(i+1)\%n}B_i$ is visible, otherwise it is a hidden surface. Naturally, the front surface is assumed to be visible and the back surface is assumed to be a hidden surface. The line segments, which do not belong to a visible surface, are hidden ones.

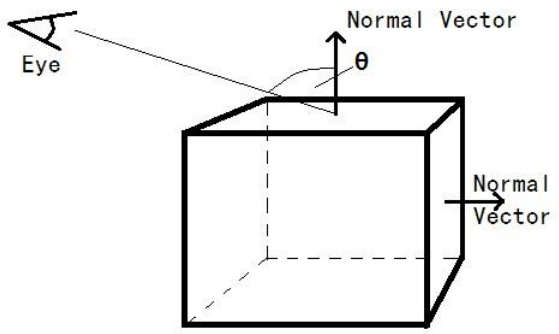

Figure 1. Elimination of Hidden Lines and Surfaces.

4. EXTENSIONS OF SVG FOR S3D MODE

Three attributes are added to support stereoscopic 3D SVG graphics, namely “isStereo”, “s3dz” and “s3dthick”, as is shown in Table 1. In this way, we can describe S3D SVG shapes in a simple way. For example, a rectangle in S3D mode with left-up coordinate of (10, 20), width of 40, height of 30, depth of 10 and thickness of 5 can be described in the following format:

<rect x=”10” y=”20” width=”40” height=”30” isStereo=”1” s3dz=”10” s3dthick=”5”/>

Table 1. Attributes added to support SVG in S3D mode.

Attribute	Value	Description
isStereo	1 | 0	S3D mode or 2D mode
s3dz	integer	depth of graphics
s3dthick	integer	thickness of graphics

5. IMPLEMENTATION ON WEBKIT

As a widely used web browser rendering engine, WebKit can parse webpages and display their contents on the screen. It parses a webpage to get a Document Object Model (DOM) tree in the first step. Another tree called “render tree” is constructed according to the DOM tree, which is composed of visible elements. The accurate position of each element is calculated in the rendering process. At last, visible elements of a webpage are painted on the screen. This main workflow of WebKit is reserved in our project.

Three important modifications are made to WebKit in our project, in order to support displaying SVG graphics in stereoscopic 3D mode.

First of all, in order to represent SVG shapes in S3D mode, additional attributes are added to each corresponding element in the DOM tree and render tree. The related attributes include the left-up position of each shape, the width and height of rectangles, the radius of circles, the semi-major axis and semi-minor axis of ellipses and the position of each vertex in both polygons and polylines. Two members are added to each element, representing these attributes in both left and right views. Furthermore, in order to generate the feeling of real S3D objects, two members are added to each element as well, representing both depth and thickness of each SVG graphics. These additional members discussed above are calculated in the rendering and layout process and used when the webpages are painted on a screen.

Secondly, in order to support S3D graphics, two frame buffers are needed. Therefore, we added two frame buffers, which are used to store the graphics representing the left and right views, respectively. As is shown in figure 2, the objects in render tree are modified, so that the calculated attributes for both left and right views can be saved in them. Meanwhile, the painting process is modified so that it can paint graphics of left view and right view respectively. The process of rendering is invoked twice to support both left and right views in our project.

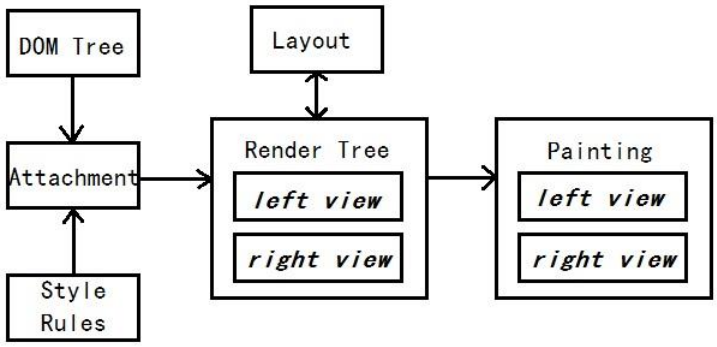

Figure 2. Modified rendering module of WebKit in our project.

Thirdly, so as to be painted on the screen, the contour of each shape is saved in an object called path in WebKit. As a consequence, we correlate two new paths, called left path and right path, to each shape element. These two paths are added to represent both left and right view graphics. They are generated in the rendering and layout process. They can be updated when the attributes of shapes are changed dynamically. When painting webpages on a screen, these paths are used for left and right views respectively.

By making the modifications mentioned above, two graphics representing both left view and right view can be generated and displayed on the screen according to an original SVG shape in a webpage.

6. RESULTS

In this experiment, SVG graphics are embedded in web pages, and the attributes and depth of these graphics are assigned when developing the webpages. We use a group of three SVG shapes as an example, which is composed of rectangles, polygons and polylines.

Figure 3 shows the original 2D graphics generated with the widely used web browser Google Chrome, which uses WebKit rendering engine as its core component. The Chrome web browser only supports the rendering and painting of 2D SVG graphics. As is shown in figure 3, the extended attributes make no difference to the web browser Google Chorme, because these unknown tags and attributes are ignored by the classical browsers.

Figure 4 shows the S3D graphics generated with our modified WebKit rendering engine, using the algorithm discussed above. As is illustrated in figure 4, visual effect of the generated graphics rendered by our modified rendering engine seems to be two graphics which are overlapped. These two graphics are the left view and right view graphics, respectively. It can be seen that there is a horizontal disparity between the left and right view graphics. Besides, the components of graphics belonging to the front surface and back surface have different disparities.

By comparing figure 3 and figure 4, it can be seen that we implemented stereoscopic 3D SVG shapes, by converting 2D SVG graphics into S3D mode, with the feeling of both depth and thickness.

In this experiment, the distance between eyes is set to be 7 cm. The resolution of viewing screen is 1920 * 1080 and the screen size of our computer is 23 inches. The position of users' eyes is assumed to be at a distance from the center of the viewing screen. The vanishing point is set to be a point far enough away from the screen and perpendicular to the screen as well. The value of the attributes representing both thickness and depth of graphics, are set dynamically when developing the web pages.

Our web browser is a modified version of Qt-port WebKit-r117946, built on the operating system of Ubuntu 12.04. The classical web browser used as a comparison in this experiment is Google Chrome.

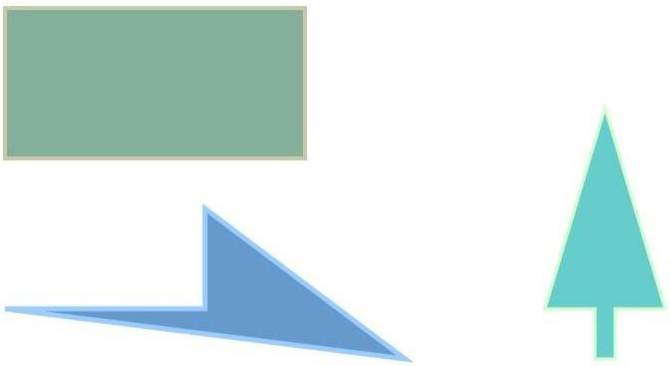

Figure 3. 2D SVG shapes generated with Google Chrome web browser.

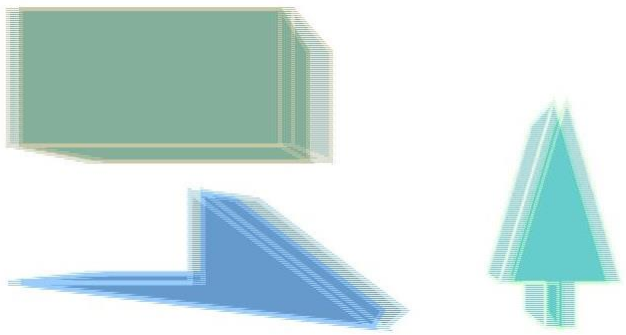

Figure 4. S3D SVG shapes generated with our modified web browser rendering engine.

7. CONCLUSIONS

This paper provides the technique to present stereoscopic 3D SVG shapes on S3D displays, by calculating both left view and right view graphics according to the original ones. 2D to S3D conversion of SVG graphics is supported with the modification of rendering module of WebKit. S3D SVG graphics are shown vividly, with a feeling of both depth and thickness, rather than simple 2D shapes. When developing webpages using our modified web browser, users can determine whether to convert 2D SVG graphics into S3D mode with an additional attribute. Meanwhile, the attributes of SVG shapes representing depth and thickness can also be set, so that different SVG graphics can be presented at different distances, seeming like real 3D objects.

ACKNOWLEDGMENTS

This project was supported by Shenzhen Special Foundation for Basic Research (JC201104210117A, JCYJ20120614150236998), by Shenzhen Engineering Lab on Intelligent Perception for Internet of Things, and by Shenzhen Engineering Lab of Three-dimensional Media Technology.

REFERENCES

[1] World Wide Web Consortium, “Scalable Vector Graphics (SVG)”, 2009, http://www.w3.org/Graphics/SVG.
[2] World Wide Web Consortium, “Scalable Vector Graphics (SVG) 1.1 Specification”, January 2003, http://www.w3.org/TR/SVG11/.
[3] Lutz, T.: SVG-VML-3D 1.3, November, 2006, http://www.lutanho.net/svgvml3d/index.htm.
[4] Jun, F., Grasso, A. , “Achieving 3D Effects with SVG” , SVG Open 2008, Nuremberg, Germany, August 26-28, (2008).
[5] Dailey, D., Elder, E.J., “A proposal for adding declarative drawing to SVG”, SVG Open 2009, October 2-4, Mountain View, CA, (2009).
[6] Wei-Yong L, Yan L. “Establish 3D Digital Elevation Model Based on SVG”, Proceedings of the 2012 International Conference on Communication, Electronics and Automation Engineering. Springer Berlin Heidelberg: 663-671, (2013).
[7] Badros G J, Tirtowidjojo J J, Marriott K, et al. “A constraint extension to scalable vector graphics”, Proceedings of the 10th international conference on World Wide Web. ACM: 489-498, (2001).
[8] Ying J, Gracanin D, Lu C T. , “Web visualization of geo-spatial data using SVG and VRML/X3D”, Image and Graphics, 2004. Proceedings. Third International Conference on. IEEE: 497-500, (2004).
[9] Paulis P. ,“3D Webpages”, Študentská vedecká konferencia , FMFI UK, Bratislava: 316-327, (2010).
[10] The WebKit Open Source Project, “How WebKit works”, http://www.webkit.org/coding/technical-articles.html.