

# Exploring Scalable Vector Graphics for Visually Impaired Users

Martin Rotard, Kerstin Otte, Thomas Ertl

Visualization and Interactive Systems Institute  
University of Stuttgart  
Universitätsstraße 38  
70569 Stuttgart  
Germany

{rotard, ertl}@vis.uni-stuttgart.de, kerstin\_otte@gmx.de

**Abstract.** Graphical information is very important in common information publishing. For visually impaired users this information is usually not accessible. Scalable Vector Graphics, a recommendation of the World Wide Web Consortium, describes graphical information in an XML document. We propose a transformation scheme into a tactile representation for this kind of graphics. Navigation modes, filters, and the output of meta information support the exploration of the graphics. Furthermore our software environment can simulate this transformation for all sizes of tactile devices.

## 1 Motivation

Nowadays it is common to have bitmap graphics on web pages. The information presented in images must be accessible to all users. For users with visual disabilities it is necessary to access this information with non-visual devices. Preparing bitmap graphics for visually impaired people in a tactile representation uses techniques like optical character recognition (OCR), because it is not possible to get any meta information like included textual content, shape type, descriptions, stroke width, etc. directly out of the graphics file [8, 10]. Fortunately the next generation of web pages will use vector graphics. The advantages of this kind of graphics are scalability, small file size and the property to store shapes, attributes, and meta information in a well defined way. Our research is based on the Scalable Vector Graphics [11] recommendation of the World Wide Web Consortium (W3C). SVG is an XML application, where shapes and attributes are encoded as plain text.

The main problems in preparing graphics for visually impaired people are the low spatial resolution and the binary mode (pins up or down) of tactile devices. Therefore graphics have to be scaled down and the colors have to be reduced to one foreground color and one background color. The additional information in SVG can be used for transformation, navigation, and filtering. The tactile output in combination with this additional information displayed on the braille line or offered by voice output makes the exploration much easier.

How to create SVG in an accessible way, can be found in guidelines of the Web Accessibility Initiative [7, 12]. Recent publications concerning accessing SVG for visually impaired people focus on specialized areas like extracting meta information [6], visualization of tactile maps [3] and transformation for tactile embosser [5].

## 2 Graphics in SVG

In SVG two-dimensional graphics are described using XML. Therefore a set of basic shape elements is defined e.g. lines, rectangles, circles, ellipses, polygons, paths, etc. The position, size, color, etc. of each shape is defined by attributes. Additionally there are special tags to group shapes `<g>` and reference self-defined symbols `<use>`. SVG provides a possibility to add alternative descriptions of the content. Therefore each element can include a title and a description tag (`<title>` and `<desc>`).

The SVG source code example below describes a series resonance circuit. Each component in the circuit is represented in a group that contains the component itself and the conductor paths. A title and a description of each group, the shape of the component, and the lines of the conductor paths represent this. In SVG 147 color keyword names are defined, e.g. *black* can be written instead of the value “`rgb(0, 0, 0)`” and *olive* instead of “`rgb(128, 128, 0)`”.

```
<?xml version="1.0"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg width="440" height="100">
  <title>Series Resonance Circuit</title>
  <desc>A resistor, capacitor and coil in series</desc>

  <g transform="translate(10,40)"
    stroke="black" stroke-width="2">
    <title>Resistor</title>
    <desc>Graphical symbol of a resistor</desc>

    <line x1="0" y1="30" x2="40" y2="30" />
    <rect x="40" y="20" width="50" height="20"
      fill="none" />
    <line x1="90" y1="30" x2="130" y2="30" />
    <text x="60" y="0" font-family="Arial"
      font-size="24">R</text>

  </g>
  (...)
</svg>
```

An SVG viewer can render the graphical representation of this circuit. Figure 1 shows the graphical output by Squiggle, the SVG viewer of the Apache Batik project [1]. On mouse over Squiggle shows the title and description of a shape or group in a tooltip.

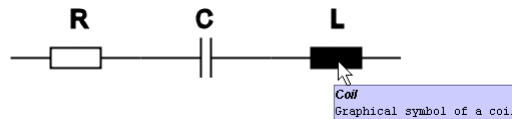


Fig. 1. Series resonance circuit in SVG

### 3 SVG Transformation Scheme for Visually Impaired People

We have developed a software environment for transforming SVG into a representation, which is accessible for visually impaired people. Our environment makes it possible to navigate in the graphics and use filters. We designed our environment to be used directly by the visually impaired users as well as a simulation environment for our development (see Figure 2).

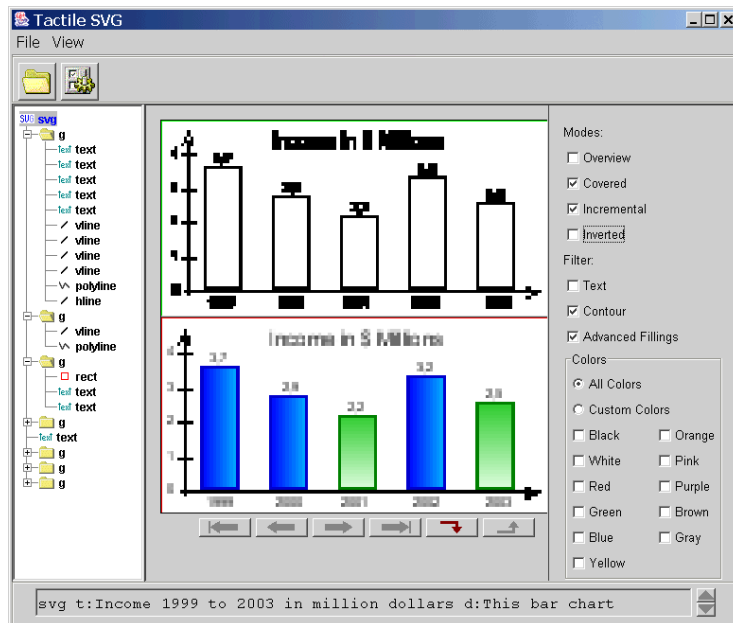


Fig. 2. Software simulation environment (scaled original graphics is at the bottom and in this example the outline filtered graphics is at the top)

The down scaled, but original graphics and the color reduced, filtered, shape separated graphics are displayed on the screen. Pins up is displayed in black and pins down in white color. There is also a simulation of a braille line, where meta data of the current shape like textual content, title, description, color, and other parameters are shown. Additionally this information is presented as voice output. For developers the shape hierarchy is displayed in a tree view. The size of the output and the length of the braille line depends on the used devices and can be configured. In our environment it is possible to zoom in/out and scroll the graphics by the user.

### 3.1 Navigation in Graphics

There is a simple concept of navigating through the shapes of a graphics. In our environment the user can navigate to the next/previous shape using the command “show next/previous element”. If the current element is a group element, it can be entered and left again. This could be done recursively in each group. There are also commands to view the first or last element in a group. One command shows the whole graphics. This can be useful to get an overview about the position of the current shape in the graphics.

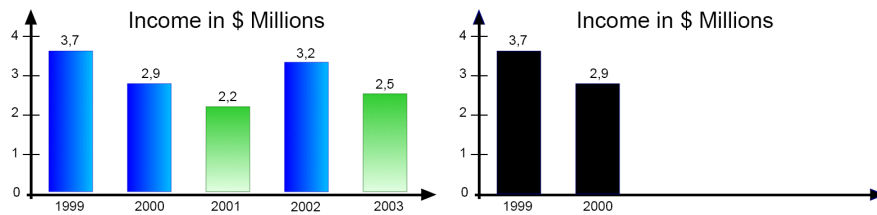


Fig. 3. Original and incremental buildup of graphics (in this case group by group)

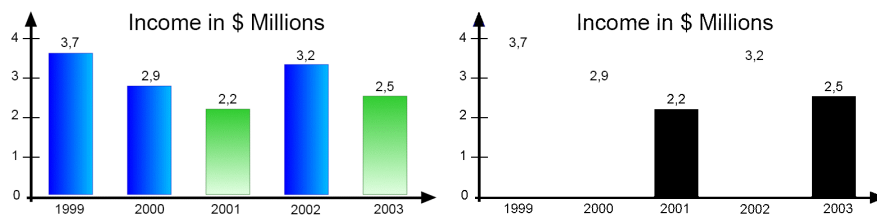
There are different modes that can help the user to explore the graphics. The buildup of the graphics can be separately shape-by-shape or accumulated incremental. Figure 3 shows the incremental buildup of the graphics. This mode can be used in groups as well, to build them up step by step. Another mode enables the user to view the elements without minding the hide rules. This makes it possible to see a car in one piece, which is partly hidden by a tree. If the user realizes, that the negative version of the output would be better, each output can be inverted.

### 3.2 Applying Filters to the Graphics

We integrated some filters to our environment that help the users to explore the graphics. Each filter can be selected in all situations and can be combined with other filters.

In our transformation scheme SVG text is extracted and presented on the braille line or as voice output. In most cases the position of the text is very important. A text filter extracts the content of text elements and shows their position on the graphics by blinking.

Filtering of colors is necessary because the output on tactile devices is binary (pins up or down). By using color filters it is possible to show just the shapes of a specific color. The color filters can be combined with color filters of an other specific color. This makes it possible to view e.g. just black and red shapes. To have a color filter for every color in the color space is not feasible. In our approach all colors (or SVG color keyword names) are mapped to a limited number of colors. For this color subset we are using the “basic color terms” established by Brent Berlin and Paul Kay [2]. These “basic color terms” consists of the colors white, black, gray, red, pink, yellow, green, blue, purple, brown and orange. We determine the mapped color by the lowest distance in the  $L^*a^*b$  color space [9]. Using color filters will show the suitable shapes with a color similar to the chosen color. If some distances are about the same, the object is shown for all of the corresponding color filters. For additional information the original color value or nearest SVG color keyword name (mentioned above) is printed on the braille line and given by voice output. A color filter of a specific color can only be selected if there is an element in this specific color in the diagram. Figure 4 shows the color filtering result of the original graphics.



**Fig. 4.** Original and color filtering result (filtered colors are black and green)

Graphics for sighted users are often hard to discretise for tactile devices. Especially gradients and textures of elements are hard to prepare for this usage in a good way. A filter removes the gradients and textures, but keeps the elements. In this case the user is informed that the gradient or texture of the current element is removed. An outline filter draws just the outlines of shapes or groups. In figure 2 the outline filtering result is shown at the top. This filter can be used to get an overview about the whole graphics.

## 4 Preliminary Results

A first evaluation of our software environment by a visually impaired colleague in our group has shown that our approaching is very promising. As equipment we used a tactile device from Metec, which has 120 to 60 tactile pins (see Figure 5). Our software environment was developed in Java. For rasterizing we used the Batik SVG Toolkit [1]. The voice output is based on FreeTTS [4]. Our system has a flexible interface, to which other tactile devices can be integrated easily. Therefore our results are not restricted to this special hardware.

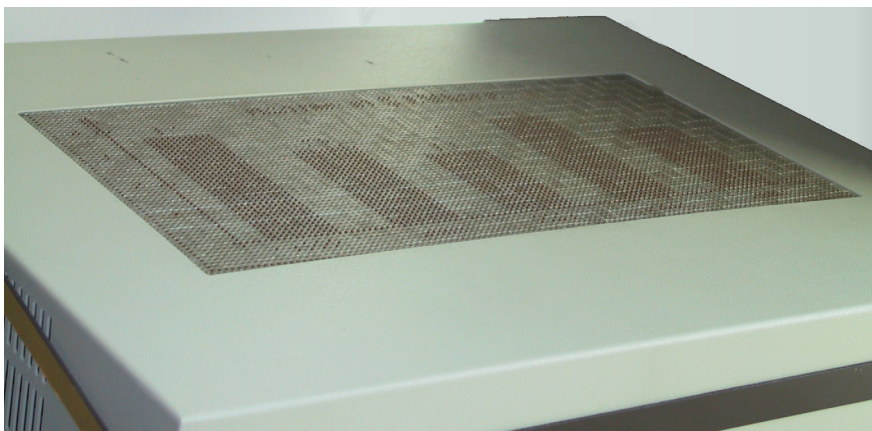


Fig. 5. Output on our tactile matrix display from Metec

## 5 Conclusion

We proposed a software environment and methods to access Scalable Vector Graphics by visually impaired people. By including all meta information like textual information (textual content, titles, and descriptions), shape type, positions, colors, and other attributes a new way of exploring graphics becomes possible. Our environment can be used as a simulation environment for the development of new extended methods in presentation of tactile graphical information.

The next version of our software environment will be able to extract SVG in HTML files. The users will be able to navigate to the next or previous SVG. The color matching in the  $L^*a^*b$  color space can be more effective if we use just the  $a$  and  $b$  component and consider the luminance separately. Furthermore it might be useful to integrate the access to bitmap graphics into our system.

### Acknowledgements

We would like to thank Waltraud Schweikhardt, Alfred Werner, Andreas Hub, and Daniel Weiskopf for their help to finish this paper and for the fruitful discussions.

### References

1. Apache XML Project: Batik, <http://xml.apache.org/batik/>, April 2004
2. Berlin, Brent; Kay, Paul: Basic Color Terms: Their Universality and Evolution. Berkeley and Los Angeles. University of California Press, 1969
3. Campin, Benjamin; Brunet, Louis; McCurdy, William; Siekierska, Eva: SVG Mapping for People with Visual Impairment, SVG Open 2003, <http://www.svgopen.net/2003/papers/svgmappingforpeoplewithvisualimpairments/>, 2003
4. Kwok, Philip; Lamere, Paul; Schröder, Marc; Vos, David; Walker, Willie: FreeTTS - A speech synthesizer written entirely in the Java programming language, <http://freetts.sourceforge.net/>, 04.04.2004
5. Gardner, John A.; Bulatov, Vladimir: Smart Figures, SVG, and Accessible Web Graphics, Proceedings of the Technology And Persons With Disabilities Conference, 2001
6. Herman, Ivan; Dardailler, Daniel: SVG Linearization and Accessibility, Proceedings of the SVG Open Conference 2002, [http://www.svgopen.net/2002/papers/herman\\_dardailler\\_svg\\_linearization\\_and\\_accessibility/](http://www.svgopen.net/2002/papers/herman_dardailler_svg_linearization_and_accessibility/), 2002
7. Jackson, Dean: Making Accessible SVG, SVG Open 2003, [http://www.svgopen.net/2003/paperAbstracts/making\\_accessible\\_svg.html](http://www.svgopen.net/2003/paperAbstracts/making_accessible_svg.html), 2003
8. Kurze, Martin: Giving Visually impaired People Access to Graphics, In: Heinz-Dieter Böcker (Ed.): Proceedings of the Software-Ergonomie '95, Workshop Nicht-visuelle graphische Benutzungsoberflächen, 1995
9. McLaren, K.: The development of the CIE 1976 (L\*a\*b\*) uniform colour space and colour-difference formula, Journal of the Society of Dyers and Colourists 92, 1976
10. Schweikhardt, Waltraud: Interactive Exploring of Printed Documents by Visually impaired People, In: J. Klaus, E. Auff, W. Kremser, W.L. Zagler (Ed): Interdisciplinary Aspects on Computers Helping People with Special Needs, ICCHP96, Linz, Austria, 1996
11. W3C: Scalable Vector Graphics, <http://www.w3.org/Graphics/SVG/>, April 2004
12. W3C: Web Accessibility Initiative, <http://www.w3.org/WAI/>, April 2004