SuperPath : a necessary primitive for vector graphics formats

SuperPath : une primitive nécessaire pour les formats graphiques vectoriels

---

Jean-Claude Moissinac
Cyril Concolato
Jean Le Feuvre

2009D018

octobre 2009

Département Traitement du Signal et des Images
Groupe MM : MultiMédia

SuperPath: une primitive nécessaire pour les formats graphiques vectoriels
SuperPath: a necessary primitive for vector graphics formats

Jean-Claude Moissinac Cyril Concolato Jean Le Feuvre
Telecom ParisTech; Institut Telecom; CNRS LTCI
46, rue Barrault
75634 PARIS CEDEX 13
{moissinac, concolato, lefeuvre}@telecom-paristech.fr

RESUME

La plupart des langages graphiques 2D utilisés sur internet aujourd'hui ne disposent pas d'une représentation efficace de la topologie des régions adjacentes. Ce manque devient problématique avec la forte croissance des applications exploitant des cartes. Dans cet article, nous présentons une nouvelle primitive, nommée superpath, pour étendre les possibilités des langages graphiques 2D existants, tells que SVG. Cette primitive permet la représentation des contours à l'aide d'un ensemble de morceaux réutilisables de contours. Nous donnons les motivations de cette proposition en les illustrant notamment sur l'exemple des cartes et avec des problèmes associés à leur édition, leur affichage et leur adaptation. Nous présentons des résultants obtenus grâce à l'utilisation de cette nouvelle primitive.

Mots-clés
cartes, adaptation, graphique vectoriel, langage multimedia

ABSTRACT

Most 2D graphics languages deployed on the internet today lack the ability to efficiently represent the topology of contiguous regions. This is becoming problematic with the growing number of map-based application. In this paper, we introduce a new graphical primitive, called superpath, to extend existing languages for 2D vector graphics, such as SVG. This primitive is a way to represent the contours in a 2D graphic with a set of reusable chunks of contour. We present the motivations for this work with examples of 2D maps and with some associated problems related to their editing, rendering, or adaptation. We give some results which could be obtained if the new primitive were used.

Categories and Subject Descriptors
I.3.3 [Picture/Image Generation] – *Antialiasing*; I.3.4 [Graphics Utilities] - *Picture description languages*; I.3.6 [Methodology and Techniques] *Languages:*

General Terms

Keywords
Map, adaptation, vector graphic, multimedia language

1. INTRODUCTION

With the increasing diversity of terminal and networks, modern multimedia technologies need to address multi-publication and adaptation of multimedia services and documents to some environment context (terminal, user preferences …). Some previous works, like [1], have shown that the more structured a document is, with rich meta-information, the greater the capability to process it, to modify or adapt it is.

In our work on multimedia documents, we usually work with common 2D graphics format such as SVG or MPEG-4 BIFS as well as other, more specialized languages (MIF, InkML...). SVG is becoming a common representation for 2D graphics document (web browsers, Linux distributions) and, with the adoption of the Tiny profile by groups such as MPEG, OMA or 3GPP, it is foreseen to be a central format on mobile devices (phone, portable media players…). MPEG-4 BIFS is used to displaying text, media and graphics for interactive services in the T-DMB broadcasting system, used in Korea for mobile TV and in France for digital radio.

Both formats rely on a similar representation of 2D vector graphics based on the use of contours with filling and stroking properties. However, these languages have limitations when editing, adapting or interacting with complex graphics content. More precisely, both languages lack the ability to define an area as a list of pieces of contour. This feature is present, for example, in the MapInfo Interchange Format (MIF/MID) [7] dedicated to map representation. In this paper, we present an extension to the SVG language called superpath, which allows defining a region as a set of contours.

The remainder of this paper is organized as follows. Section 2 describes related works. Section 3 presents the motivations of this work and explain the benefits of the superpath approach. Then, Section 4 proposes a syntax for the SVG language of the superpath extension. Section 5 evaluates the proposal and finally Section 6 concludes this paper and indicates future work.

2. RELATED WORKS

The following works and technologies described in this section are related to our proposal.

In [6], Gangnet & al. present the benefits of representing a 2D graphics by a structure were all the regions are disjoined and only

share frontiers. They call this structure a planar map. The advantage of such representations is a complete knowledge of the topology of a drawing, enabling powerful processing and rendering. However, the main drawback of this approach is a complex data structure, not suited for usage on mobile devices. Moreover, in our scenario, such a structure is not well suited, because the concepts used are too far from the one available in SVG or MPEG-4 BIFS

In a compromise between the planar map representation and the contour representation, the Flash format [Reference] defines Shape objects. Shape objects are the smallest graphical entity that can be manipulated: either for transformations or for user interaction. The content of a Shape object is a sequence of lines and curves, for which a color to the left, a stroke color and a color to the right are defined. This representation enables a very efficient conversion to bitmap and solves some of the problems mentioned in this paper (e.g. antialiasing and format compactness). On the other hand, there is no notion of region within a Shape, and points within a Shape cannot be reused in another Shape. It is therefore quite difficult to maintain the semantics of original regions in a Flash content.

In [4], Dailey proposes new elements in the SVG language. One of them has some common part with our approach. A major benefit of our proposal compared to this work is to preserve structural information in a simple way, closer to the SVG design. We have furthermore followed a language-agnostic approach, applicable to other languages.

3. RATIONALE OF SUPERPATH

In this section we will first formulate the interest of such a primitive through an example. We then highlight a series of benefits of its adoption.

3.1 The map example

Figure 1 A map with reuse of some geometrics elements

Figure 1 represents some of the departments and regions of France. The regions are composed of several departments. Thus the borders of each department belong to two polygons, those in neighboring departments, and also to the polygons that define the regions. Finally, it is common that a department border is defined by the course of a river. Some contour chunks will be redefined up to five times in the same way: two for contours of the departments, two for the contours of a region, one for the course of a river.

Such a representation is usual for a map; two neighbor contours commonly share a piece of contour. Existing languages for 2D vector graphics such as SVG cannot currently express this relationship which leads to problems in several applications associated to graphics from editing, to adaptation or to rendering. The following section details the benefits of using a more adequate representation.

3.2 Benefits of superpath

Benefit 1 - To enable coherent adaptation

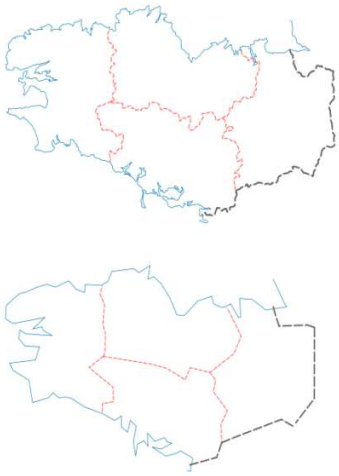

Figure 2 Coherent adaptation

If the SVG file contains the knowledge that two chunks of contours are the same, both contours could be adapted in coherent manner, possibly applying only one processing (Figure 2) and avoiding artifacts produced by divergent adaptations (Figure 3).

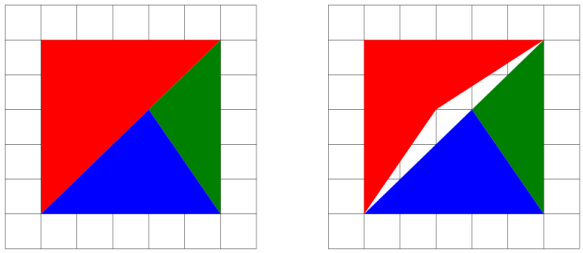

Figure 3 Artifact due to divergent adaptation of the diagonal on different contours

Benefit 2 - To facilitate edition of the contours

If the SVG file contains the knowledge that two chunks of contours are the same, an editor could suggest to the user either to modify both in the same manner, or to do different modifications.

Benefit 3 - Managing the antialiasing problem

The rendering model of SVG, commonly shared by other technologies, presents a problem with regards to anti-aliasing as illustrated in Figure 4. The SVG rendering model assumes that drawing on the canvas follows the painter's algorithm, i.e. objects encountered first in the document are drawn first, and objects following in the document are rendered on top. Imagine that the blue region is rendered first against a white background. Its border will be anti-aliased using a mix of white and blue. Then, when the green region is rendered, its borders will be anti-aliased as a mix of green and previously rendered (white+blue) pixels. Hence, instead of having a mix between green and blue, the result is a mix between white, blue and green. In such case, knowing that the border is common between the green and blue region, it would be possible to produce a better anti-aliased result.

Benefit 4 - To reduce the file size

For a lot of vector graphics (e.g. maps, cartoon drawings...), the main part of the data reside in the representation of the points which define the contours. When group of points are duplicated, the file size is artificially augmented. Using a representation taking advantage of the structure of the points would therefore reduce the file size.

Holes are contours inside a contour which define region excluded from the main contour (and presumably transparent). Using a better representation of duplicated chunks implies better representation of holes.

Benefit 5 - To optimize the display

A player which knows that two chunks of contours are the same could optimize the rendering, for example performing only once the linearization of Bezier curves into straight segments.

More generally, it is useful and simple to preserve the semantic of chunks identity for various processing covering a wide range of scenarios.

4. SUPERPATH IN SVG

This section presents a solution to the problems aforementioned and then details the proposed syntax for the SVG language.

4.1 Principles

Our approach can be applied to any vector graphics format using a contour-based representation of objects. There are generic principles to follow for these languages where the superpath could be added.

Defining chunks

We need to introduce the concept of chunks using the target language model. As a chunk can be used by different contours with different display style, a chunk shall not be dependent on a style. The chunk therefore only contains geometrical data (points and control points).

Using chunk references

We need to be able to reuse these chunks in different shapes. This is usually achieved through explicit or implicit identifier for the chunk and further referencing in each shape which uses this chunk. Obviously, if the targeted language has syntax to express references, we need to be coherent with that syntax for chunk referencing (e.g., SVG xlink or BIFS DEF/USE mechanisms).

Reversing a chunk order

As illustrated in Figure 4, duplicated chunks could be used as there are first defined or in the reverse order. The syntax must enable such a usage.

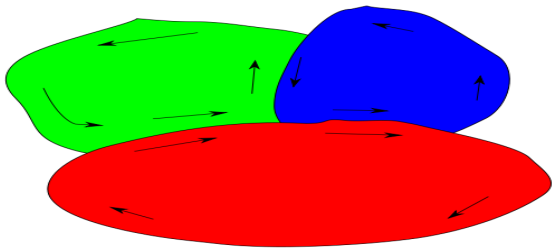

Figure 4 Three shapes and the walk along each contour

Defining chunk joins

We need to define how two successive chunks of a path are joined. The simplest way is to draw a line between the end of a chunk and the beginning of the next.

4.2 Proposed syntax for SVG

In this section, we present a syntax of superpath for the SVG language which is compact and easy to author. We propose the following new SVG elements:

- <superpath>: this element has no specific attribute. It supports traditional SVG attributes such as id, transform, class … It is used as a container element for the elements below and defines its own styling.
- <subpath>: this element declares a chunk used in a superpath. Its syntax is the same as the standard SVG <path> element, but the semantic is different. This element is only displayed when it is a child of a <superpath> element.

A subpath can have two forms:

- It can contain a d attribute, analog to the one in the path element; if it has an id attribute, then the d part can be reuse by reference in another superpath;
- It can contain a xlink:href attribute which references the id of another subpath. In this case, its geometry is defined by the d attribute of the referenced subpath..

In addition, the geometry of a subpath can be used in the reverse order, as explained in 4.1. A “order” attribute is therefore defined with possible values ‘direct’ or ‘reverse’; the ‘direct’ value is the default.

In both forms, the “cmd” attribute is used to express how the subpath is joined with the previous one. There is currently only one possible value “line” indicating that a straight line shall be used to join the chunks.

We can notice that the definition of a chunk can be similar to the definition of the corresponding part of the original content. An overhead will come from the size of the references to the chunk.

As an example, Figure 4 can be represented as follows:

```
<superpath id="redpath" fill="red">
  <subpath id="sp1" d="M...." />
  <subpath id="sp2" d="...." />
  <subpath id="sp3" d="...." />
</superpath>
<superpath id="greenpath" fill="green">
  <subpath id="sp4" d="M...." />
  <subpath id="sp5" d="...." />
  <subpath xlink:href="#sp1" />
</superpath>
<superpath id="bluepath" fill="blue">
  <subpath xlink:href="#sp2" />
  <subpath id="sp6" d="...." />
  <subpath xlink:href="#sp4" order="reverse" />
</superpath>
```

5. EVALUATION

As explained previously, the superpath approach gives great benefits in many use cases (adaptation, interactivity). We must however check the compactness of the proposed representation.

We have implemented a program which takes an SVG file, finds all the <path> elements, computes all reused chunks and then produces an SVG file extended with the superpath/subpath elements.

We can note that when the original content has no path or when no chunk of path is reused, our algorithm does not modify the content. For other cases, our algorithm produces files in which semantic information about regions is restored. We present below some results for significant files.

Table 1. Test set

File name (.svg)	File size	SVG path	Points	Duplicated points	Cumulated path size
MapFrance	119 Kb	336	9452	5434	81 Kb
Toon1	103 Kb	50	21179	12961	101 Kb

As we can see in table 1, these samples –representative of our tests- are mainly composed of path data. Therefore, a better representation of path data is likely to result in a gain in size.
We can see some results in table 2. The first column is the name of the original file. The second column is the size in kilobytes of that file. The third column is the size of the generated file using the <superpath> element.
The files used are SVG files taken from the web (maps from Wikipedia) or from previous projects (extract from cartoons [2][3]).
As we can see, a gain in size is observed in many test content, and in other cases the increase in size is quite low..
The gain could be really greater if the file were properly build. As an example, it is common with hand-made maps that the frontier between is present in two <path> elements with slight differences due to the authoring tool. In such cases, our algorithm cannot currently determine that the points are semantically identical, and the resulting file size is therefore not as compact as it could be if the content were properly authored.

Table 2

Graphics (.svg)	Original	With superpath
MapAfrica	642	571
MapFrance	873	851
Toon1	103	105
GAEcho	102	55
G0005	36	34

6. CONCLUSION

We have shown that a superpath feature is really useful for modern vector graphics formats. We have also descibed how such a feature could be introduced in these languages, and provided syntax for the SVG language. This syntax has been tested on several sequences and generally provides a gain in size.

In future work, we will further improve the structure for a better compactness of the representation and investigate other chunks join types. We plan to propose this feature to the W3C SVG Working Group, and to propose a similar feature in MPEG-4 BIFS.

7. REFERENCES

[1] Bulterman D.A., Specification and Support of Adaptable Networked Multimedia, 1993, Multimedia Systems 1(2), 68 - 76.

[2] Concolato C., Moissinac J.C., Dufourd J.C., Representng 2D cartoons using SVG, SMIL Europe 2003

[3] Concolato C., Dufourd J.C., Moissinac J.C., Dufourd J.C., , Creating and encoding of cartoons using mpeg-4 bifs: methods and results, IEEE Transactions on Circuits and systems for Video Technology.

[4] Dailey D., Suggestions for additions to the W3C Specification for SVG, http://srufaculty.sru.edu/david.dailey/svg/spec.html

[5] Flash File Format, www.adobe.com/devnet/swf/pdf/swf_file_format_spec_v10.pdf

[6] Gangnet M., Hervé J.C., Pudet T., Van Thong J.M., Incremental Computation of Planar Maps, ACM SIGGRAPH Computer Graphics Volume 23 , Issue 3

[7] MIF File Format, www.**mapinfo**.com/free/docs/mipro/*mipro_70_users.pdf*

Dépôt légal : 2009 – 4ème trimestre
Imprimé à l'Ecole Nationale Supérieure des Télécommunications – Paris
ISSN 0751-1345 ENST D (Paris) (France 1983-9999)

TELECOM ParisTech

Institut TELECOM - membre de ParisTech

46, rue Barrault - 75634 Paris Cedex 13 - Tél. + 33 (0)1 45 81 77 77 - www.telecom-paristech.frfr

Département TSI