Journal of Computers Vol. 28, No. 1, 2017, pp. 181-192
doi:10.3966/199115592017022801014

An VRIS Representation Method Based on XGML

Jing Qiu[1*], Jia-zheng Yuan[2], Hong-zhe Liu[2], and Cheng Zhou[1]

[1] Beijing Union University, Beijing, China
qiujing681@163.com

[2] Beijing Key Laboratory of Information Services

Received 24 September 2015; Revised 17 February 2016; Accepted 11 April 2016

Abstract. In this paper, we have proposed a method of VRIS representation base-XGML. We have designed an imitation XGML model based on SVG and XML to assisting our VRIS representation. In the processing of our method, convert image features into a meta-data descriptor and using a split-and-merge algorithm transform image into element descriptors by utilizing XGML model. The model was applied to an experimental data set which included various vector images. Experiment results shows that our VRIS representation method can efficiently represent raster images which originally were vector image to a semi-structured XGML document, and it succefully reduced storage capacity of image and increases the speed of image retrieval.

Keywords: semi-structured, SVG, XGML, XML

1 Introduction

With the rapid advances in network application technology, there has been a significant increase in the number of raster images which were vector image originally (Vector-Raster image) on the internet. Vector-Raster image take up large amounts of storage capacity and exist in the form of unstructured data (irregular or unclear), making it difficult to search them. The eXtensible Graphics Markup Language (XGML) imitates the rules of XML and SVG, transforming image storage into text collection instruction systems with certain formats and structures. The Vector-Raster image semi-structural (VRIS) format refers to an image being stored as XGML-based instructions. This enables XGML to be easily searched and further reduces storage requirements. However, because images are composed of a series of discrete and irregular pixels, the VRIS representation encounters some difficulties.

Extensible Markup Language (XML)[1] is a standard format which establishes a structure for documents and data, allowing for more accurate searches and a more convenient file transfer. The creation of XML offers possibilities for storing images with a definite format and structure. Scalable Vector Graphics (SVG) [2] is a new generation two-dimensional vector graphics standard format created by W3C and based on the rules of XML in network applications. Its dynamic graphical description format based on text could better facilitate network transmission and information retrieval. Due to the advantages of SVG, which are able to meet the requirements of user browsing, it has been widely applied to multimedia data processing in WebGis [3-4], 3G mobile phones [5-7], data statistics [8-9], and other fields. Simultaneously, the SVG sharing resource Web [12] established by SVG researchers exhibits several advantageous applications. Furthermore, the commercial software package VectorEye has the potential to transform images into SVG documents.

As part of this study, we have developed a simple yet effective XGML model based on the rules of XML and SVG, which is suitable for the description of raster Vector-Raster image. Then applied image content and human visual perception to extracted image features and transformed it into f metadata descriptions in the XGML model. In addition to this, images were divided into different regions by using a split-and-merge algorithm and acquired regions were then transformed into image element descriptors in the XGML model. Experiment results shows that the method used in this paper could better transform

* Corresponding Author

Vector-Raster image into semi-structured XGML documents, and it successfully reduced storage capacity and increases the speed of image retrieval.

2 Related Work

This study pertains to data processing based on XML and SVG. Several authors have made advances in this field in recent years.

The most unique feature of the XML language is that it enables users to create a mark which could meet their own application requirements. This means that XML is no longer text piled with various kinds of labels and elements, but it is a systematic structure with application value [13-14]. At present, there are numerous authors who have integrated XML and databases so as to strengthen flexibility and scalability, including relational [20], object-oriented [19], and object-relational [24]. In order to solve for the uncertainty and inaccuracy of ambiguous information, XML has been applied to the representation and reconstruction of such information. For example, Abiteboul [16] established a display system based on XML. Oliboni et al. [25] provided a detailed explanation of the ambiguity of data in the XML document, which provides a model-transforming relational data set to XML text.Li et al. [15] rebuilt the nested fuzzy relational database and established the Fuzzy XML data model. Peres et al. [26] proposed an XML structure to represent fuzzy information, from which they could construct an XML structure according to the type of data, so as to realize the function of the fuzzy search. Furthermore, XML is beneficial for data retrieval. For instance, Hong et al. [13] developed an intelligent image retrieval system(IIRS) based on XML text. By representing multi-layered metadata structures, displaying local characteristics of images, global features, and the meaning of image data, IIRS makes image retrieval based on content and meaning possible. Azzam et al. [14] proposed an image indexing and retrieval system (CIIRX) based on the content of XML documents. Having determined the features found in images, CIIRX is able to exact the description of this feature to generate an image index by MPEG-7 structured annotation standards. It is then able to realize image retrieval based on the generated XML Index documents. This rule of XML has also been applied to image compression [10] and image understanding [11].

SVG is able to describe three kinds of entities: vector graphs, images, and text. SVG is a kind of dynamic and interactive language, which supports stretching transformations under different resolutions. In 2004, Byron [21] introduced a method to transform raster images to XML and SVG text. This method records the RGB and coordinate values of each pixel point and transforms raster images into XML documents using the XML rule. Finally, they are converted to SVG documents using the SVG rule. Even though this method can transform simple images to XML documents and SVG documents respectively, it requires transforming all pixels in every image, which results in a low switching rate and XML and SVG documents with large capacities. Byron [21-22] has made a more detailed elaboration of the potential of transforming the image of raster geographic information to XML and SVG documents using XML code. This could transform the image of raster geographic information into an SVG document with a series of HTML collections, finally transforming the SVG document to GML by XSLT.

This method makes the descriptions of ranks and regional collections more accurate. The method described in [21-22] divides images into different regions according to a single pixel color and then extracts the edges of different regions in order to describe image regions and edge information using SVG code. However, this method has a low generation rate of SVG documents and features an obvious marginal sawtooth after enlarging the image.

Therefore, Yuan et al. [18] put forward a novel method which divides images into different regions and extracts the edges. After this step, the images were merged, stretched, and closed using polygon fitting (Literature [2-22] did not introduce edge fitting for the images), finally describing it by the rule of SVG. Although Yuan's [18] method simply establishes an edge fitting based on [21-22], his method greatly improves the quality of the generated SVG vector graph.

In the context of prior research, this paper proposes the concept of a semi-structuralization of the image. Based on the features of XML and SVG, this paper has designed a simple but effective XGML model which is suitable for the description of the semi-structured images. During this study, image feature information was extracted and then transformed into metadata descriptors in the XGML model. In addition, the image was divided into different regions using a split-and-merge algorithm, which then transformed the acquired regions into image element descriptors in the XGML model. Experiment results shows that the method used in this paper could better transform Vector-Raster image into semi-structured

XGML documents, and it successfully reduces storage capacity and increases the speed of image retrieval.

3 XGML Model Design

XML can express different features and image content, while the instructions in XML can achieve convenient image retrieval (Fig. 1 gives a description for tree structures in XML documents). SVG is a standard format for two-dimensional vector graphics (Fig. 2 is an image description example for SVG, Fig. 2(a) is an original vector image, Fig. 2(b) is a code for SVG).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE image>
<image>
   <!Describe the image's whole feature>
   <Name> Image_Name </Name>
   <Keyword> Image_Key Message  </Keyword>
   <Link> Image Address  </Link>
   <Size> Image_Size  </Size>
   <Shape> Image_Shape  </Shape>
   <ColorPattern> Image_Color  </ColorPattern>
   <OtherFeature>
        Image_Feature
   </OtherFeature>
   <!Describe the first subobject's feature>
   <Subobject1>
      <Position>
          <CoordinateX>
            Subobject1_CoordinateX
          <CoordinateX>
          <CoordinateY>
            Subobject1_CoordinateY
          <CoordinateY>
      </Position>
        ⋮
      <
      <Subobject11>
      <!Describe the first subobject's subobject feature>
       ⋮
      <Subobject11>
  </Subobject1>
   <!Describe the Nth subobject's feature>
   <SubobjectN>
    ⋮
  </SubobjectN>
</image>
```

Fig. 1. An example of an XML document

```
<?xml version="1.0" standalone="no"?>                    Header file of SVG
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN"
        "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
<svg  xmlns=http://www.w3.org/2000/svg
        xmlns:xlink="http://www.w3.org/1999/xlink">
   Monchrome Definition of image Region
  <style type="text/css">
  <!CDATA[
   .color1{fill:247,141,55 stroke="none"}
   .color2{fill:255,255,255 stroke="none"}
  .color3{fill:0,0,0stroke="none"}
     ......
  ]]>
  </style>
  Gradient Definition og Image Region
  <defs>
     <LinearGradient id="region1">
        <stop offset="0%" stop-color="#eb6f7a" />
          .....
         <stop offset="100%" stop-color="#e76d78"/>
     </LinearGradient>
       ........
   </defs>
  <g id="Tigger" widh="460" height="740" x="0" y="0">
     <!Feature of Gradient Region>
     <path  style="fill:url(#region1)" d="Mx0 y0 Cx1 y1..."/>
     <!Feature of MonochromeRegion>
     <path id="region2" class="color2" d="Mx0 y0 Cx1 y1..."/>
     <path id="region3" class="color3" d="Mx0 y0 Cx1y1.../>
     ......
  </g>
  </svg>
```

(a) (b)

Fig. 2. An example of SVG; (a) representation of a tiger; (b) SVG code for the tiger

In order to utilize the advantage of SVG and XML, we have designed a concise and efficient eXtensible Graphics Markup Language (XGML) model which is suitable for VRIS presentation. The XGML model includes two collections: (1) Image Metadata Descriptor (IMD); (2) Image Element Descriptor (IEM). Fig. 3 is a framework of the XGML model.

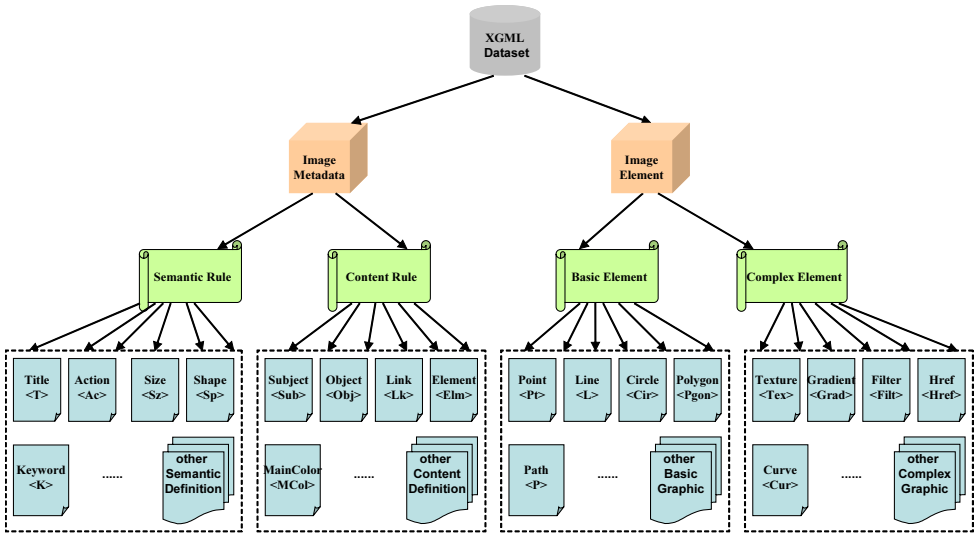


Fig. 3. The frame Structure of an XGML System

3.1 Image Metadata Descriptors

In different areas Metadata has various definition, the widely definition is “data about data”. In this paper, the IMD is mainly using to describe the content and semantic of image’s information. We now present one of the primary components of the Image Metadata Descriptor in the XGML model. The instructions in XML can achieve convenient image retrieval. Thus, we reference XML grammar to design an IMD which uses semantic and content information to describe images. The IMD is briefly divided into two parts: (1) Semantic Rules and (2) Content Rules.

(1)Semantic Rule: resulting from the image itself, it's mainly using to describe the apparent and location information It is beginning with "<>" and ending with "</>". Fig. 4(a) displays a collection of semantic rules. We will introduce some common rules as follows:

<T>is used for describing an image's title, beginning with<T>and ending with </T>. If there has a sub item in image, we can using <SubT> </ SubT >acts as a sub-item of <T>.

<Ac>is designed to describe action information in an image object, it has many attributes to present specific actions of the object in an image, such as: Play, Hit, Touch, etc. We can describe the information between <Ac> and </Ac>.<Sz>is designed to describe size information for the image; it includes an attribute, such as: height,width,coordinate, and so on.

<Sp>is used to describe shape information for the image; it includes an attribute, such as sharpen, smooth, etc.

(2)Content Rule: resulting from scene and vision information, mainly describing information of the image feature. As the same as the Semantic Rule, it is always paired, beginning with "<>" and ending with "</>".. Fig. 4(b) features a collection of content rules; we briefly explain some frequently-used objects here:

<Sub>is designed to describe subject information in the image; it includes an attribute, such as: target, color, and so on.

<Obj>is designed to describe object information in the image; it includes an attribute, such as: target, color, and so on.

<Elm>is designed to describe information which differs for the subject and object but for some location and reason for activating events in the image; it includes an attribute, such as: target, color, and so on.

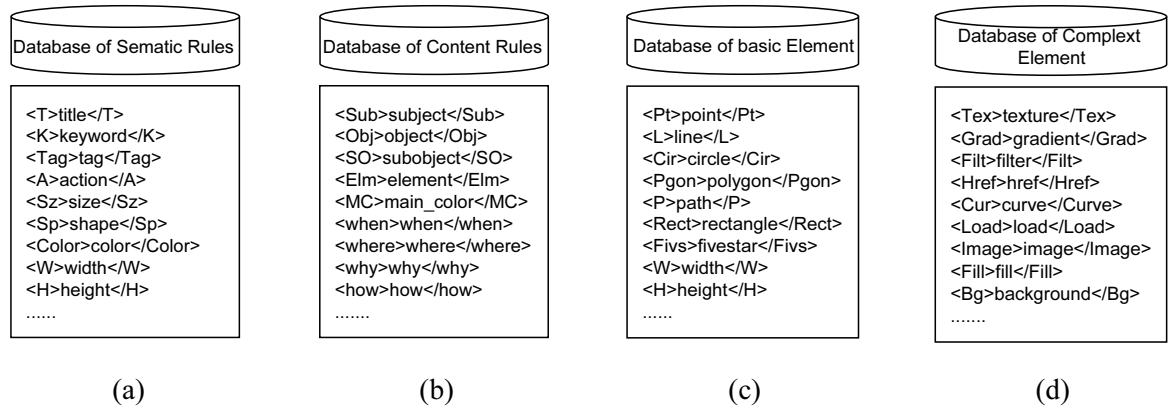


Fig. 4. (a) A collection of basic semantic rules. (b) A collection of basic content rules. (c) A collection of basic graphics. (d) A collection of complex graphics.

3.2 Image Element Descriptor

We now explain the next key component, the Image Element Descriptor in the XGML model. Binary raster images consist of pixels, which causes the storage capacity to increase. SVG grammar uses a different form to describe the image region in reducing the storage capacity. While certain flags in the SVG grammar are fairly long, they do provide additional storage capacity. Therefore, we reference SVG grammar in designing an IED that can meet requirements and clearly express the meaning of an image. The IED is primarily used to represent different shapes in the image region, which can decrease image storage space. It is mainly divided into two types: (1)Basic Element Object; (2)Complex Element Object.

(1)The Basic Element Object type is similar to the basic shape in SVG [2] ; Fig. 4(c) shows a collection of basic element objects. We briefly recommend some frequently-used objects below: point, line, and polygon.

<Pt> point is used to describe pixel information, including coordinates "x" and "y," color "color," etc. As shown bellow:

<Pt x="x" y="y" style="fill: rgb(color)"></Pt>

<L> is designed to describe different kinds of line segments. The "style" attribute is used as a definition of line type, such as "Line" for a straight line and "PolyLine" for a poly-line. <L> also includes a

starting point “S”, a few turning points “T”, and an ending point “E”; it also includes other attributes such as color, line width, etc. The example showing as bellow:

<L style=“Line/Polyline” points=“Sx0 y0 Tx1 y1 x2 y2…Ex y” color=“RGB” stroke-width =”strokewidth”/>

<Pgon> is used to describe different types of shapes, such as: rectangle “Rect”, triangle “Trangle”, five-pointed star “FiveS”, etc. It uses “position=” to define the angle’s coordinates and uses “style” to define the attributes of a polygon.

<Pgon points=“x1 y1, x2 y2, x3 y3…”Style=“fill:r,g,b”…. />

(2)The Complex Element Object is a complicated command which can’t be represented using basic element objects; Fig.4(d) shows a collection of complex element objects. We introduce some common objects here:

<Grad> describes a smooth transition process from one color to another. <Grad> includes a starting coordinate (x1, y1), an ending coordinate (x2, y2), a starting color (r1, b1, g1), and an ending color (r2,b2,g2), etc. By modulating those attributes we can achieve different gradient processes, such as horizontal gradient and vertical gradient, etc. The following example is the linearGradient:

<Grad x1=“0%” y1=“0%” x2=“100%” y2=“0%”>

<stop offset=“0%” style=“stop-color:rgb(r1,g1,b1)”/>

<stop offset=“100%” style=“stop-color:rgb(r2,g2,b2)”/>

</Grad><href> is used to describe a complex image component which can’t be expressed using a document.

<href url=“complex.jpg”/>

<css> is similar to a style sheet in HTML[3], we can write a reused style into style sheets.

<css type=“external-style/css” link=“External.css” />

4 Method of VRIS Presentation

This section mainly describes the processing of converting an image into a semi-structured XGML model. As described in section 4.1, we extract the image semantic information and content information based on image contextual information and visual information and then exchange them to a metadata descriptor. The target of this part is in order to make image retrieval more convenient; Fig. 5 is a pipeline illustrating the image metadata extraction process. Section 4.2 describes a process that changes binary image data into image element descriptors through a series of transformations. The intention of this part is to reduce the image storage capacity; Fig. 6 shows the main process.

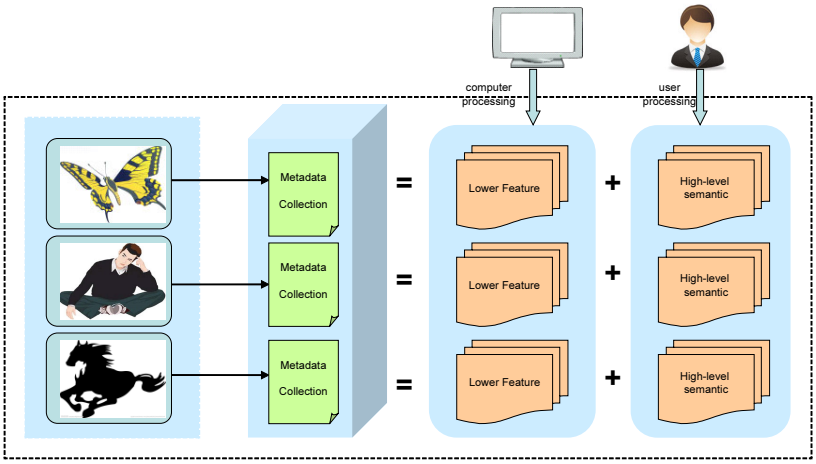


Fig. 5. The processing of metadata extraction 4.1 Converting Image Information into Metadata Descriptors

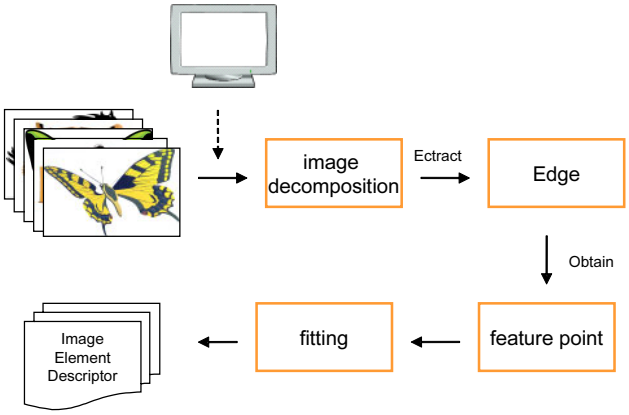
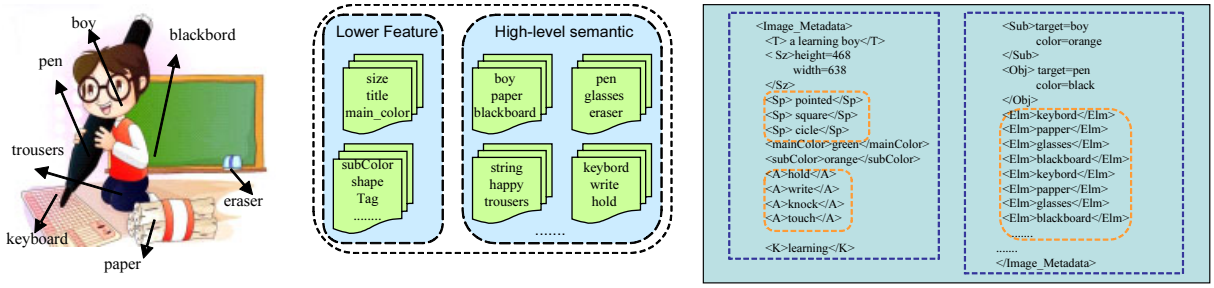


Fig. 6. A pipeline of image element conversion

IMD extraction includes the extraction of semantic data pertaining to the description of an image's appearance and content data containing a description of the scene and vision information. Setting M as a primitive data collection, $S \in M$ is the semantic data in the image, $C \in M$ is the content data of the image, and $M = \{S, C\}$. Because of internal discreteness and complexity of an image, it is difficult to acquire extraction image metadata automatically, so we use man-machine interactive methods to extract metadata.

By using a computer to automatically extract lower-level visual features and then passing user data to joining high-level semantics information, we can achieve the goal of extracting image metadata. The lower-level features we extracted mainly included color, texture, and shape features. The high-level features marked by the user are based on visual perception and subjective recognition used to annotate the different image scenes. Finally, we combined the lower-level features and the high-level semantics into the XGML model. Fig. 7 is an example of extracting metadata, Fig. 7(a) is the input image, Fig. 7(b) is the main data information from (a), Fig. 7(c) is the document description metadata from (a).



(a) (b) (c)

Fig. 7. (a)An input image. (b) The Extracted Metadata from (a). (c) An XGML file transformed from (b).

4.2 Converting Image Pixels into Image Element Descriptors

This section mainly introduces the process of converting a Vector-Raster image into an IED in the model XGML through a series of transformations. The first step in our algorithm is to input the Vector-Raster image, and then we conduct a region division-merge operation and extract the covered area edge information using the Canny [17] algorithm. Lastly, we convert this Vector-Raster image into an IED in the model XGML.

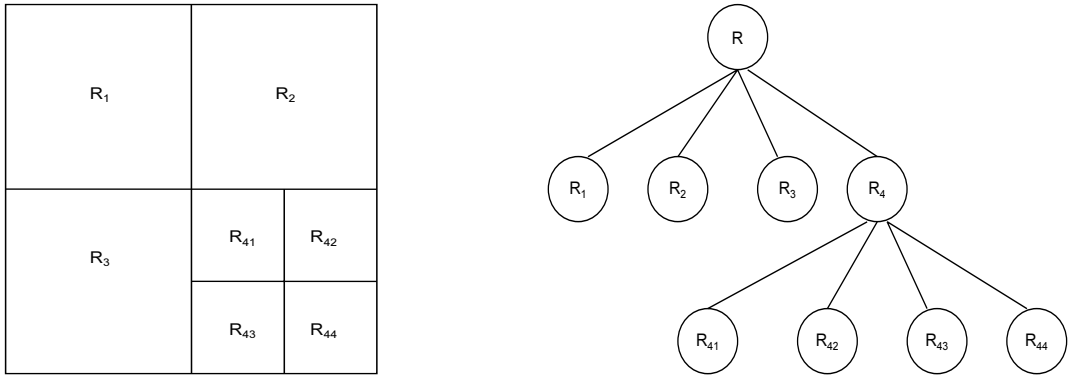
Processing of Region Segmentation. The image element conversion process firstly requires image region segmentation. Image division is divided into two steps: region division and region merging. The region division and merger are mainly based on the consistency of the region. When different image characteristics exist in a region, the region will be divided into four equal sub-regions. When neighboring sub-regions satisfy the characteristics of consistency, they will be merged into a larger region. After the image division is finished, the region will be decomposed using edge, shape, and color features. The specific steps are as follows:

(1) Initially, set A as an image space, $I \in A$ as a binary raster image, and R as the region of the whole image. The R computation is based on the domain decomposition method, the concrete split steps are outlined as follows:

① Calculating the gray average of the region R_i : $m_i = \frac{1}{N_i} \sum_{x \in R_i} f(x)$ (N_i is the number of pixels in R_i, $f(x)$ is the gray value of x).

② Judging each pixel and gray average of the region R_i whether it satisfies the formula $\max_{x \in R_i} |f(x) - m_i| < \xi$ (ξ is the selected threshold). If it does satisfy the result, $P(R_i) = true$; if not, the $P(R_i) = false$.

③ If $P(R_i) = false$, the region R_i will be divided into four neighboring sub-regions: $R_i = \bigcup_{k=1}^{4} R_{ik}$ (Fig. 8). If $P(R_i) = true$, the region R_i will stop dividing.



(a) (b)

Fig. 8. The process of region segmentation. (a) A partitioned image. (b) The corresponding quad-tree.

④ The four sub-regions R_{ik} repeat the steps ① to ③ until the sub-regions stop dividing.

(2) If neighboring sub-regions R_i and R_j (each object may have a different size) can satisfy $P(R_i \cap R_j) = true$, the neighboring sub-regions are integrated into a total region m_i .

The Edges of Region. The edge information $e(m_i)$ in the region m_i , included a large number of unneeded pixels which increased the storage space of the image. We reduced the edge information storage space by extracting the edge feature points.

Normally, the direction of corner points in the digital image edge curve has significantly changed. For the edge information $e(m_i)$ in region m_i , the main idea of this method is that using a_i to indicate the direction of point i-1 to i , the curvature d_i of a_i to a_{i+1} can be used to determine whether this point is a feature point or not. The main steps are shown below as follows:

(1)Set the current edge points which need to be calculated as i , and a_i as the direction of points i-1 to i , a_{i+1} is the direction of point i to $i+1$.

(2)Let d_i be the curvature of a_i to a_{i+1} . If $|a_{i+1} - a_i| < 4$, $d_i = |a_{i+1} - a_i|$; if $|a_{i+1} - a_i| > 4$, $d_i = |a_{i+1} - a_i| - 8$; if $|a_{i+1} - a_i| = 4$, $d_i = 4$. So $d_i = \{0, \pm 1, \pm 2, \pm 3, 4\}$.

(3)If $|d_i| = 0$, it means this point lies on a straight line, so the point is not a feature point; if $|d_i| > 2$, it means the point is a feature point and its coordinates need to be recorded (x_i, y_i) . If $|d_i| = 1$ or 2 , then the point is a doubting point and its coordinates (x_i, y_i) need to be recorded.

(4) We repeat steps (1) to (3) until the collection of all edge feature points is determined

$F\left(e\left(m_i\right)\right)=\overline{P_{i_1}P_{i_2}...P_{i_n}}$ in the region m_1.

We extracted image edge feature points by calculating the directional change in the neighboring pixels; this reduced the storage space of the image edge and is a steppingstone to the next step in the image element transform.

Transforming to Image Element Descriptors. We have obtained a feature point collection $F\left(e\left(m_i\right)\right)=\overline{P_{i_1}P_{i_2}...P_{i_n}}$ by employing the above method. For the purposes of creating image resolutions, we used the method in [9] detailing the fitting process for different kinds of edge feature points in the images. These were then transformed into image element descriptors in the model XGML.

If the slope $K_{ij}\left(P_{ij}\right)$ value for $\forall P_{ij}$ in $F\left(e\left(m_i\right)\right)=\overline{P_{i_1}P_{i_2}...P_{i_n}}$ is the same constant σ (Fig. 9(a)), this region edge $e\left(T_i\right)$ can use $\overline{P_{i_1}P_{i_n}}$ for fitting onto a straight line $Line\left(e(T_i)\right)$. If the slope $K_{ij}\left(P_{ij}\right)$ value for part of $\forall P_{ij}$ in $F\left(e\left(m_i\right)\right)=\overline{P_{i_1}P_{i_2}...P_{i_n}}$ is different (Fig. 9(c)), this region edge $e\left(T_i\right)$ can use $\overline{P_{i_1}P_{i_2}...P_{i_n}}$ for fitting onto a curve $Curve\left(e(T_i)\right)$ (Fig. 9(c)). If slope $K_{ij}\left(P_{ij}\right)$ for some part of $\exists P_{ij}$ in $F\left(e\left(m_i\right)\right)=\overline{P_{i_1}P_{i_2}...P_{i_n}}$ contains the same constants υ, this region edge $e\left(T_i\right)$ can use $\overline{P_{i_1}P_{i_m}P_{i_s}...P_{i_n}}$ for fitting onto a $Polyg\left(e(T_i)\right)$.

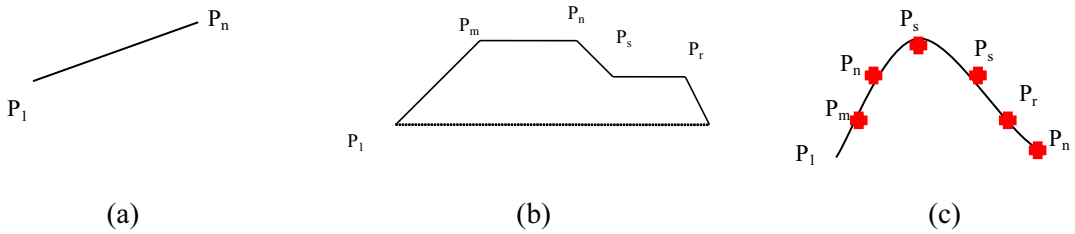


Fig. 9. (a) A straight line. (b) A curve. (c) A fold line.

Next,we will be describing a transform image to image element descriptor in the XGML model after completion of the areas m_i, specific steps are outlined as follows:

①If the edge $e\left(T_i\right)$ type is linear $Line\left(e(T_i)\right)$, we use the following XGML code indicators:

<L style="Line" points="Sxi1 yi1 Exin yin" color="ri,gi,bi" stroke-width ="strokewidth"/>

②If the edge $e\left(T_i\right)$ type is polyline $Line\left(e(T_i)\right)$, we use the following XGML code indicator:

<L style="Polyline" points="Sxi1 yi1 Txi2 yi2 xi3 yi3…..Exin yin" color="ri,gi,bi" stroke-width ="strokewidth"/>

③If the edge $e\left(T_i\right)$ type is polyline $Line\left(e(T_i)\right)$, we use the following XGML code indicator:

<L style="Polyline" points="Sxi1 yi1 Txi2 yi2 xi3 yi3…..Exin yin" color="ri,gi,bi" stroke-width ="strokewidth"/>

④If the edge $e\left(T_i\right)$ type is curve $Line\left(e(T_i)\right)$, we use the following XGML code indicator:

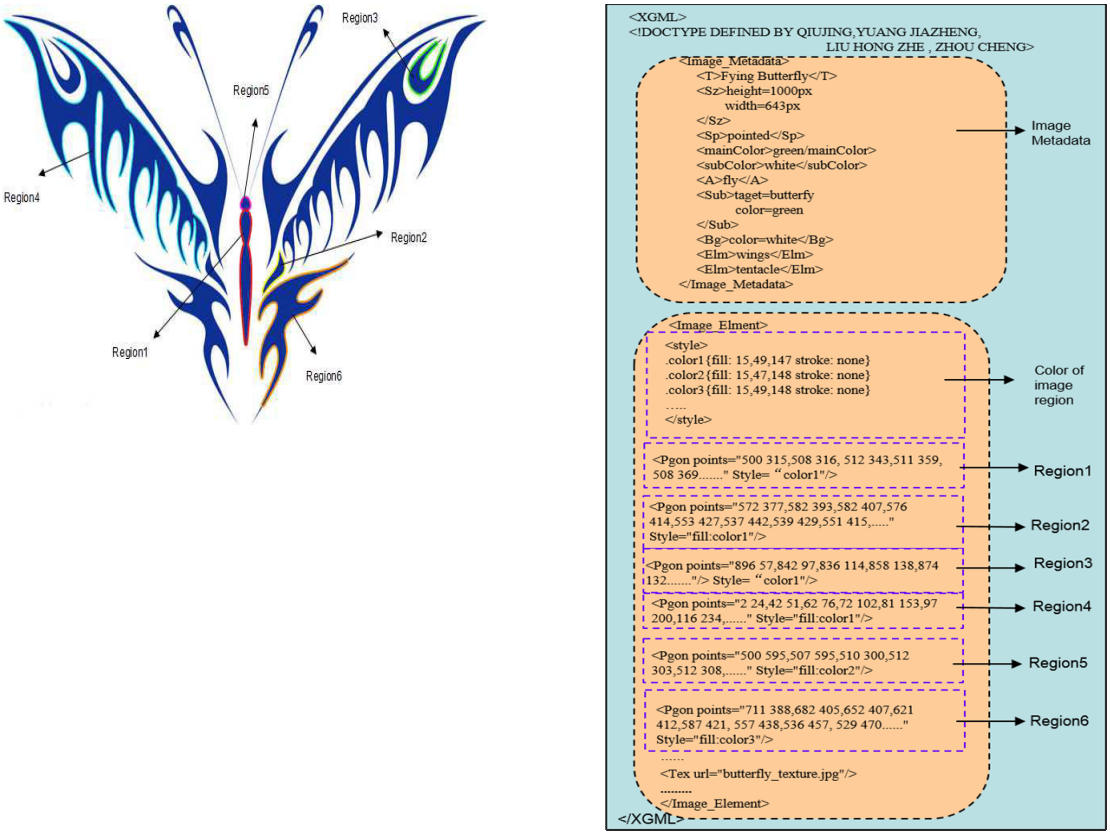
<path p="Mxi1 yi1Cxi2 yi2Ci3yi3….Cxin yin" stroke=" ri,gi,bi" stroke-width ="strokewidth"/>

5 Results and Discussion

We have fully developed the representation method for VRIS based on an XGML model, which was tested on Vector-Raster image. The experiment was based on building the XGML model and identifying the target for transforming the Vector-Raster image to semi-structured documents. The semi-structured documents can be conveniently used in image retrieval and reduce the required image storage space.

Fig. 10(a) shows an input Butterfly Vector-Raster image. Fig. 10(b) is the part code display of Butterfly images which were transformed into a semi-structured XGML model. The data further illustrates our semi-structured image algorithm, based on XGML, in Tables 1 and Table 2. This technique is convenient for image retrieval and reduces the required image storage space. As shown in Table 1, for the tiger image, butterfly image, horse image, and thinking-man image, we obtained results by comparing the origi-

nal image title number and the amount of metadata information after the semi-structured documents. From Table 1, we observe that the semi-structured documents transformed by our method were more convenient for the purpose of image retrieval than the original image. In Table 2, we respectively compared the original storage capacity of the tiger image, butterfly image, horse image, and thinking-man image, with the storage capacity after they were converted to XGML semi-structured documents. From Table 2, we observe that the semi-structured documents transformed by our method featured storage capacities which were smaller than the original images.



(a) (b)

Fig. 10. (a) The original butterfly image. (b) The XGML document for (a).

Table 1. Comparison of quantity for image information between original method and proposed method

image	Information for original image (number)	Information for proposed VRIS method (number)	Differ (number)	variation
Tiger	4	7	3	1.75
Butterfly	3	9	4	3.0
Horse	4	8	4	2.0
Thinking-man	3	10	7	3.3
Flower	3	9	6	3.0
Grass	4	11	7	2.75

Table 2. Comparison of capacity for original image and transformed XGML document

Image	capacity for original image (kb)	capacity of proposed VRIS method (kb)	Reduced (kb)	Reduced Amplitude(%)
Tiger	127	119	8	6.299
Butterfly	130	124	6	4.615
Horse	53	50	3	5.7
Thinking-man	29	26	3	10.3
Flower	78	69	9	11.5
Grass	103	95	8	7.77

6 Conclusion

In this paper, we have introduced a semi-structured image method based on XGML. Our approach is primarily intended for Vector-Raster image on the basis of constructing an XGML model. Experiments conducted with different Vector-Raster

image demonstrated that our method can efficiently express Vector-Raster image in a semi-structured XGML document which reduces storage capacity and is convenient for image retrieval. Our method also has certain limitations, Such as, the XGML instruction system we designed is not yet ideal and will require a more in-depth study and the semi-structured object is a new concept in this field and will require more in-depth understanding and analysis. In the future, we will continue to enrich the concept of semi-structured objects and build a more perfect XGML model.

Acknowledgments

The authors would like to thank anonymous reviewers for their valuable comments. We would also like to thank Jia-Zheng Yuan and Hong-Zhe Liu for their kind help. This work was supported by the following projects：The National Natural Science Foundation of China (No.61271369);Beijing Natural Science Foundation (4152018), The National Key Technology R&D Program(2014BAK08B02),The Project of Construction ofInnovative Teams and Teacher Career Development for Universities and Collegesunder Beijing Municipality (CIT&TCD20130513), Funding Project for Academic Human Resources Development in Beijing Union University (BPHR2014E02).

Reference

[1] Extensible-Markup-Language (XML). <http://www.w3.org/XML/>

[2] Scalable Vector Graphics (SVG). <http://www.w3school.com.cn/svg/>

[3] P. Wang, L.Y. Dong, Web GIS research and application based on SVG, Applied Mechanics and Materials 513(2014) 2004-2007.

[4] H. Huang, Y. Li, G. Gartner, Y. Wang, An SVG-based method to support spatial analysis in XML/GML/SVG-based Web-GIS, International Journal of Geographical Information Science 25(10)(2011) 1561-1574.

[5] M. Yuan, J.S. Yuan, G. Huan, D.D. Wang, The mobile oilfield map based on SVG and information Integration, Advanced Materials Research 798(2013) 349-352.

[6] F. Lin, C. Guo, Raster-vector integration based on SVG on mobile GIS platform, in: Proc. of Pervasive Computing and Applications (ICPCA), 2011.

[7] Q. Zhao, F. Deng, W. Zhang, X. Zeng, The research of mobile GIS power distribution line inspection based on mobile SVG/J2ME, Physics Procedia 24(2012) 1038-1043.

[8] F. Molina, B. Sweeney, T. Willard, A. Winter, Building cross-browser interfaces for digital libraries with scalable vector graphics (SVG), in: Proc. of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries, 2007.

[9] R. Kunze, Dynamic and interactive visualization of weather data with SVG, in: Proc. of SVG Open 2005 Conference and Exhibition, 2005

[10] D. Maheswari, V. Radha, Secure layer based compound image compression using xml compression, in: Proc. of Computational Intelligence and Computing Research (ICCIC), 2010.

[11] D.-Z. Zhao, W. Li, J.-Z. Yang, An XML-based process definition language for medical image understanding, in: Proc. of Computer Application and System Modeling (ICCASM), 2010.

[12] The Graphical Web. <www.SVGOpen.org>.

[13] S. Hong, Y. Nah, An intelligent image retrieval system using XML, in: Proc. of Multimedia Modelling Conference, 2004.

[14] I.A. Azzam, A.G. Charlapally, C.H.C. Leung, J.F. Horwood, Content-based image indexing and retrieval with xml representations, in: Proc. of Intelligent Multimedia, Video and Speech Processing, 2004.

[15] W. Li, X. Chen, Z.M. Ma, Reengineering fuzzy nested relational databases into fuzzy XML model, in: Proc. of Fuzzy Systems (FUZZ-IEEE), 2014.

[16] S. Abiteboul, L. Segoufin, V. Vianu, Representing and querying XML with incomplete information, in: Proc. of 12th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, 2001.

[17] J. Canny, A computational approach to edge detection (IEEE, Trans.), Pattern Analysis and Machine Intelligence, 8(1986) 679-714. (Canny edge detection)

[18] J.Z.H. Yuan, Y.J. Wang, H. Bao, An effective method of presentation of raster image based on vector graphics SVG, Chinese Journal of Electronics 1(2008) 188-193.

[19] T. Naser, R. Alhajj, M.J. Ridley, Flexible approach for representing object oriented databases in XML format, in: Proc. of the 10th International Conference on Information Integration and Webbased Applications Services, 2008.

[20] G. Kappel, E. Kapsammer, W. Retschitzegger, Integrating XML and relational database systems WWW Journal, 7(2)(2004) 343-384.

[21] B. Antoniou, L. Tsoulos, Converting raster images to XML and SVG-the potential of XML encoded images and SVG image file in geometrics. <http://svgopen.org/2004/papers/Converting_raster_images_ to_XML_and_SVG/>, 2004.

[22] B. Antoniou, L. Tousles, The potential of XML encoding in geometrics converting raster images to XML and SVG, Computers & Geosciences 32(2)(2006) 184-194.

[23] M.A. Levine, R.E. Marrs, J.R. Henderson, D.A. Knapp, M.B. Schneider, The electron beam ion trap: a new instrument for atomic physics measurements, Physica Scripta T22(1988) 157.

[24] M.J. Carey, J. Kiernan, J. Shanugasundaram, E. Shekita, S. Subramanian, XPERANTO: middleware for publishing objectrelational data as XML documents, in: Proc. of the 26th International Conference on Very Large Databases, 2000.

[25] B. Oliboni, G. Pozzani, Representing Fuzzy information by using XML schema, in: Proc. of the 2008 19th International Conference on Database and Expert Systems Application, 2008.

[26] F.F.F. Peres, R. dos S. Mello, A rule-based conversion of an object-oriented database schema to a schema in XML schema, in: Proce. of the 4th IEEE International Conference on Digital Information Management, 2009.