SVG Rendering of Real Images Using Data Dependent Triangulation

Sebastiano Battiato
University of Catania
battiato@dmi.unict.it

Giovanni Gallo
University of Catania
gallo@dmi.unict.it

Giuseppe Messina
STMicroelectronics
giuseppe.messina@st.com

Abstract

This paper presents a novel technique to convert raster images in a Scalable Vector Graphic (SVG) format using Data Dependent Triangulation (DDT). The triangulation, a classical 3D graphic rendering approach, is here applied to digital images acquired by imaging consumer devices. Good quality rendering of real images has been obtained making use of some ad-hoc heuristics able to properly manage advanced SVG features (e.g. path, gradient, filter effects). Experiments and comparisons with existing techniques confirm the effectiveness of the proposed strategy.

Keywords: Data Dependent Triangulation, SVG, Imaging devices.

1 Introduction

This paper presents a novel technique to transform digital raster images into SVG format. The SVG standard allows representing complex graphical scenes by a collection of graphic vectorial-based primitives, offering several advantages with respect to classical raster images such as: scalability, resolution independence, etc. To vectorize a digital raster image, one has to understand, at some level, the underlying semantic content of the scene in order to recognize shape, plot, colors, etc. Several published methods [2],[8] aimed at 3D graphic rendering of real or artificial scenes introduce solutions that guarantee good quality at the price of a high computational cost. In this work we are interested in finding some heuristic techniques to cover the gap between the graphical vectorial world and the raster real world typical of digital photography in a specific application; SVG format could find useful application in the world of mobile imaging devices, where typical camera capabilities should match with limited color/size resolutions displays.
We use the method known as Data Dependent Triangulation (DDT) [6] to approximate local pixel neighbourhood by triangles. Recent advances in such field can be found in [9] where using simulated annealing a lower number of triangles is achieved. DDT has been also successfully applied in digital image interpolation ([14], [16]) outperforming classical techniques.
Further advanced techniques for image polygonization ([7], [8]) are available, but for real-time processing, only solutions with a minimum computation overload are allowed [1]. Such triangulation could be directly managed by SVG primitives. Although the quality achieved in this way is rather good the size of the resulting files may be very large. Taking into account local redundancies (e.g. flat areas) the amount of data coming out from initial DDT maybe sensibly reduced. Preliminary results show good performances in terms of compression ratio, already choosing simple merging techniques as reported below in the experimental section. Some attempts to increase visual quality of converted SVG images by properly using gradient primitives will be reported at conference meeting.
The proposed technique has been compared with other raster to vector conversion methods ([12],[15]) showing good performances mainly in terms of both perceptual and measured quality.

The paper is structured as follows. The next section summarizes the main capabilities of the SVG standard, reporting also an overview of its intrinsic potentiality. Section 3 presents the underlying ideas of the Data Dependent Triangulation technique while in section 4 the main steps of the proposed technique are presented. The successive section reports some visual results of the proposed techniques together with some useful comparisons with existing approaches. A conclusions section closes the paper tracking directions for future research and works.

2 Scalable Vector Graphics

SVG standard is a World Wide Web Consortium (W3C) recommendation designed for many purposes (e.g. advertising, clip art, business presentations, etc.) also using advanced features like animation and filter effects. The language describes 2D-graphics and graphical applications in XML, and it is mainly devoted to a series of application where scalability in terms of output resolution, color spaces, and available bandwidth is required.
The core of the current SVG developments is the version 1.1. Actually the SVG 1.2 specification is under development and available in draft form. SVG Mobile

Profiles (Basic and Tiny) targeted to resource-limited devices, are part of the 3GPP platform for third generation mobile phones. Finally SVG Print is a set of guidelines to produce final-form documents in XML suitable for archiving and printing. SVG is a drawing tool having as features all the basic vector graphics primitives. These include lines, polylines, polygons, rectangles, circles, and ellipses. On top of these basic shapes, SVG also supports paths, including cubic and quadratic Bezier curves and elliptical arcs. It is also possible to specify transformations on objects either as an additive instruction (scale, translate, rotate, or skew) or as a global transformation matrix. Major details can be found directly at the W3C site [13] while in ([5],[11]) the overall SVG capabilities are summarized.
In this work we are mainly interested in unsupervised conversion into SVG format of real images acquired by typical handset imaging devices.

3 Data Dependent Triangulation

DDT (Data Dependent Triangulation) technique is able to outperform classical methods such as nearest neighbour, linear/cubic interpolation, using a piecewise linear intensity function. The input image is replaced with a set of triangles according to a specific cost function able to implicitly detect the edge details (e.g. high frequencies) as described in [16]. The overall perceptual error is then minimized choosing a suitable triangulation. Recently further optimization of the cost function has been introduced in [14] for color demosaicing of CFA (Color Filtering Array) Bayer data.
The triangulation problem is described as follows. Given a polygonal area Ω in the <*x-y*> plane, and a set $V=\{(x_i,y_i)\}$ where $i=1,2,..,N$ of not collinear distinct points contained in Ω; let f be a non-degenerated function, defined in Ω but unknown:

$$f(x_i, y_i) : \Omega \Rightarrow \Re , \qquad (1)$$

therefore the values assumed by f in the N selected points are

$$f(x_i,y_i) = F_i \qquad i=1,2,\dots,N \qquad (2)$$

A "triangulation" on Ω is a set $T=\{ T_i \mid i=1,2,..,q\}$ of non-degenerated triangles that satisfies the following conditions:

1. Each triangle vertex is an element of V, and each element of V is a triangle vertex;
2. Each edge of a triangle in T contains exactly two points of V.
3. T is a partition of Ω, therefore

$$\Omega = \bigcup_{\forall T_i \in T} (T_i) \qquad (3)$$

$$T_i \cap T_j = \varnothing \, ; \forall T_i, T_j \in T : i \neq j. \qquad (4)$$

In the Ω image space the function f can be, for instance:

a. The grayscale image (for grayscale space);

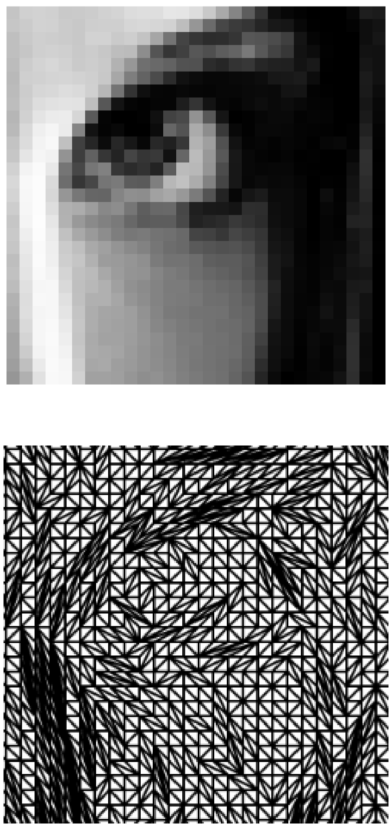


Figure 1 - A magnified detail of classical "Lena" image with relative triangulation.

b. The luminance of the three channels R,G and B in the color images.

V contains the known pixels position set of the matrix, and the F_i are their respective values. Then the data collected are:

$$V_i=(x_i,y_i,F_i) \, ; \, Pixels(x_i,y_i)= F_i \quad \text{for} \quad i=1,2,\dots,N \qquad (5)$$

This collection of information permits to work in the 2.5D grayscale terrain space by fixing the image area Ω, lying on <*x-y*> plane, and as the third axis, the function f applied over the points of the <*x-y*> plane.
The resulting regular mesh has a vertex triangle at every pixel: for a given image with MxN pixels a 2xMxN triangle list is built, plotting two triangles for each pixels (as visible in Figure 1), where the diagonals are locally chosen to match the image edges.

In figure 1 a magnified detail of classic "Lena" image with relative triangulation is shown.

In this paper DDT conversion is used as a first conversion step; starting from the "pixel world" an intermediate representation is built such that the underlying structure of the image itself is intrinsically represented by the final triangulation.

4 The Proposed Technique

4.1. Triangulation

Our technique is mainly based on [16]. To obtain a vector-graphics representation of a raster images I of size MxN, a rough initial triangulation is first achieved in a $2M$x$2N$ grid using classic Delaunay triangulation (equiangular).

The image reconstruction process, by using an iterative method should assign low cost for edges (high texture details) representing visually relevant parts. The dependency of the edges from the function $f(x,y)$ is assumed to be only relative to the four vertices forming the quadrilateral created from the two adjacent triangles (Figure 2). The edge swapping, modifies the internal diagonal according to some local property. We used also a look-ahead expansion of the local optimality, making use of an accurate analysis of local neighbourhood (8 vertices) . As described in [16] the adopted cost function (Sederbergs cost function – SCF) for the edge between triangles T_1 and T_2 is

$$\text{cost}(e) = \|\nabla P_1\| \cdot \|\nabla P_2\| \cdot (1 - \cos\theta) \qquad (6)$$

where θ is the angle between the contour-line normals P_1 and P_2, $\nabla P_i = (a_i, b_i)$ is the gradient on the plane P_i, and a_i, b_i are the coefficients; therefore the gradient magnitude is obtained by: $\|\nabla P_i\| = \sqrt{(a_i^2 + b_i^2)}$.

Recognizing that the directions of the contour normals and the gradient are identical, we can simplify this to

$$\text{cost}(e) = \|\nabla P_1\| \cdot \|\nabla P_2\| - \nabla P_1 \cdot \nabla P_2 \qquad (7)$$

that is

$$cost(e) = \sqrt{(a_1^2 + b_1^2)} \cdot \sqrt{(a_2^2 + b_2^2)} - (a_1 \cdot a_2 + b_1 \cdot b_2) \qquad (8)$$

In our case, to further speed-up the overall process we used the following approximation:

$$cost(e) = (|a_1| + |b_1|) \cdot (|a_2| + |b_2|) - (a_1 \cdot a_2 + b_1 \cdot b_2) \qquad (9)$$

that is cost effective. Such cost function is similar to the simplified cost function used in [6] to describe the roughness of triangulated terrain (Sobolev seminorm). Further investigation will be devoted to compare the cost function (9) with alternative approaches as described in ([6],[14],[16]).

The global triangulation cost is summarized in:

$$cost(T) = \sum_{\forall t_1, t_2 \text{ contiguous in } T} \left[(|a_1| + |b_1|) \cdot (|a_2| + |b_2|) - (a_1 \cdot a_2 + b_1 \cdot b_2) \right] \qquad (10)$$

The DDT approach aims to minimize the global triangulation cost by using local edge-swapping together with simple look-ahead optimization (see [16] for major details). A few iterations are needed to guarantee an effective coupling between original edges distribution and related triangle vertices map. In our working case, image conversion from a raster to SVG, using consumer imaging device, three/four iterations are usually sufficient to guarantee effective results. Actually, after the fourth iteration the triangulation cost changes slightly introducing, for our visualization purposes, just little improvement.

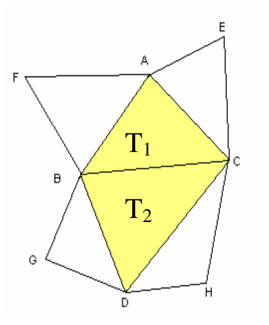


Figure 2. The figure shows how the local triangle structure could be arranged according to the cost function used by the DDT.

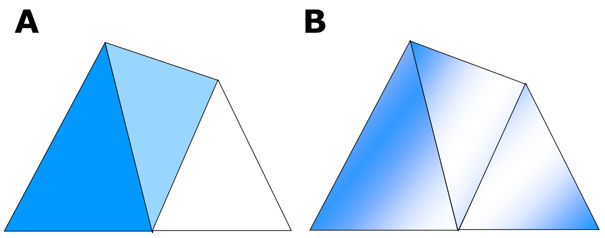


Figure 3. The use of smart filling rules, by involving linear or radial gradient, permits a further improvement of the resulting triangulation. In **A** only the mean value of the three vertex colour has been assigned, in **B** an estimation of linear gradient has been simulated.

4.2. Basic SVG Rendering

After DDT each triangle can be easily remapped into a SVG representation in the following way:

```
<path d="M x1,y1 L x2,y2 L x3,y3 Z"
stroke="#FFFFFF" fill="#FFFFFF"/>
```

where `x1,y1`, `x2,y2`, `x3,y3` are the vertex coordinates, and *#FFFFFF* is the filling color, obtained at this stage as the simple mean value of the corresponding pixel grey levels. The generalization to RGB images is simply obtained, replacing *#FFFFFF* with *#RRGGBB*, where *RR, GG, BB* are respectively the hex representation of the red, green and blue mean value inside the considered triangle.

The collected *2xMxN* triangles, described as simple "text" can be properly "zipped", by classic Lempel-Ziv algorithm, obtaining an SVG compliant file. Although the perceived quality is already good, the overall amount of bit size requested by such approach is very high.

4.3. Rendering Refinement

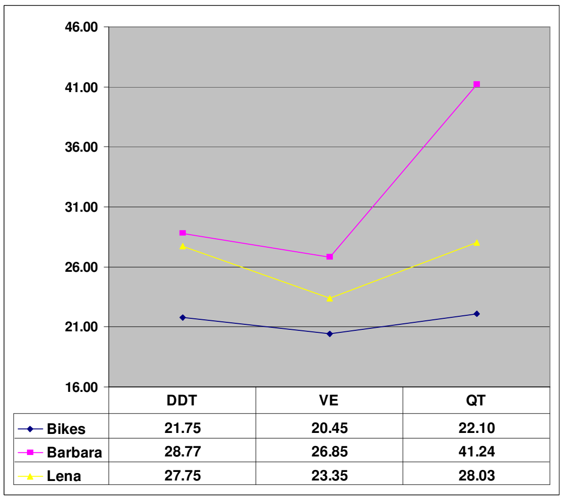
Adjacent triangles are merged together according to some similarity properties. As first approach we merge together adjacent triangles, whenever their corresponding filling colour is almost the same. Using this simple heuristic rule a gain improvement of about *20%* in terms of overall bit size is obtained.
Advanced techniques as described in [2],[4],[10] will be considered to further refine the preliminary data-dependent triangulation reducing at the same time the number of SVG primitives. An effective polygonization ([7], [8]) coupled with smart filling rules able to recognize “graphic structure” (e.g. linear or radial gradient) directly captured by SVG primitives are also under investigation (see Figure 3).

5 Experimental Results

The final quality of SVG rendering, coming from raster images has been assessed through an extensive experimental phase. Experiments have been conducted both on a typical standard dataset of images and on a database of images acquired by real handset imaging devices to whom the proposed technique is mainly oriented. For sake of comparison, SVG images obtained by simple quad-tree decomposition [12] and results provided by commercial software [15] are reported.
Figures 4, 5 and 6 show the resulting outputs, together with a 400% magnified detail, of three standard images “Lena”, “Barbara” and “Bikes” (Figs. 4.a, 5.a, 6.a) rendered in SVG by using the Vector-Eye software [15] (Figs. 4.b, 5.b, 6.b), the proposed strategy (Figs. 4.c, 5.c, 6.c) and the quad-tree decomposition (Figs. 4.d, 5.d, 6.d). The overall perceived quality, (the texture details, sharpness, etc.) is sensibly improved. Quad-tree decomposition [12], introduces unpleasant blockiness, also using very relaxed splitting rules. Vector-Eye [15] draws a sort of “draft” summary of the original image, dropping many details.
As objective quality metrics we use PSNR (peak signal-to-noise ratio); PSNR is very common in image processing (e.g. comparison between an original image and a coded/decoded image). Given a source image *f(i,j)* that contains *N* by *N* pixels and a reconstructed image *F(i,j)* where *F* is reconstructed by decoding the encoded version of *f(i,j)*, PSNR measured in decibels (dB) is computed by using:

$$PSNR = 20\log_{10}\left(\frac{255}{\sqrt{MSE}}\right) \qquad (11)$$

where MSE is the Mean Square Error between *f* and *F*, computed on pixel basis. Typical PSNR values range between 20 and 40.



	DDT	VE	QT
Bikes	21.75	20.45	22.10
Barbara	28.77	26.85	41.24
Lena	27.75	23.35	28.03

Table 1. PSNR results of 3 standard images, after SVG conversion obtained by the proposed method (DDT), VectorEye software (VE), quad-tree decomposition (QT).

Table 1 confirms effectiveness of the proposed method, reporting the measured PSNR with respect to the original raster images. The quad-tree approach gives a higher PSNR but this result is only due to the nature of quad-trees implementation: each square is filled with the corresponding mean gray level, smoothing all details.

In effect the main purpose of our approach is the improvement of the input image quality obtained by relaxing the constraints between “pixel world” and resolution. At this stage we are mainly interested in high quality rendering versus scalability. By simply zooming the resulting SVG images (see magnified detail of Figs. 4, 5 and 6) the enhancement introduced by DDT is visible. The overall extent minification/magnification where the method works well could be easily verified using a typical SVG viewer (typically 4/5 times the original size). Considering the Lempel-Ziv algorithm, provided by SVG format, the overall compression ratio obtained by the proposed strategy is about 2:1 on the average; also in this case an effective improvement will be obtained introducing some advanced rendering features.
The different resolution size of relative sensor with respect to the viewfinder display in mobile devices is a relevant issue. Figure 7 shows SVG rendering of color images acquired by using an Evaluation Kit [17] (STV6500 - E01 EVK 502 VGA). The first two images (Figure 7.a and 7.b) have been captured with a QVGA format (160x120 pixels) typically used in mobile phone devices; the last image (Fig. 7.c) has been captured in CGA format (320x240 pixels) typically used in PDA devices. The SVG images have been directly rendered doubling the original resolution.
Further experiments can be found at the following web address: *http://www.dmi.unict.it/ ~battiato/svg/*

6 Conclusions and Future Works

A novel technique able to convert raster images in a SVG format has been presented. The DDT methodology is used to obtain a “draft” vector representation, refined by

simple merging heuristics. Experiments confirm the effectiveness of the proposed strategy.
More powerful mesh simplification methodologies ([2], [4]) and advanced image segmentation algorithm [3], will be analysed in order to better manage the ideal trade-off between the real appearance of raster images with respect to "vector" world. We plan to further improve the proposed method considering advanced SVG primitives (e.g. filter effects, gradients, etc.). Also the possibility to work directly on Bayer data images will be investigated.

Acknowledgements

The authors are deeply grateful to Advanced System Technology AST Catania Lab of STMicroelectronics for providing motivations to this research and constructive critics to an early version of this algorithm.

References

[1] S. Battiato, A. Castorina, M. Guarnera, F. Vella. *A Light Viewfinder Pipeline for Consumer Devices Application* – In Proceedings of IEEE *ICME'03 International Conference on Multimedia and Expo 2003*, Vol. I, Baltimore USA – (2003) pages 681-684

[2] P. Cignoni, C. Montani, R. Scopigno. *A Comparison of Mesh Simplification Algorithm.* Computer & Graphics, Pergamon Press, Vol. 22(1) (1998) pages 37-54

[3] D. Comaniciu, P. Meer. *Mean Shift: A Robust Approach Toward Feature Space Analysis*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.24 (5) (2002) pages 603-619

[4] E. Danovaro, P. Magillo, M.M. Mesmoudi. *Level-OF-Detail (LOD) Modelling of Scalar Fields Through Topology-driven Simplification*. In Proceedings of Eurographics – Italian Chapter (2003)

[5] D. Duce, I. Herman, B. Hopgood. Web 2D *Graphics File Format*. Computer Graphics *forum* vol.21(1) (2002) pages 43-64

[6] N. Dyn, D. Levin, S. Rippa, *Data Dependent Triangulation for Piecewise Linear Interpolation,* IMAJ. Numerical Analysis, vol.10 (1990), pages 137-154

[7] L. Hermes, J.M. Buhmann. *A Minimum Entropy Approach to Adaptive Image Polygonization*. IEEE Transactions on Image Processing, vol.12 (10) (2003) pages 1243-1258

[8] D. Kalvin, R.H. Taylor. *Superfaces: Polygonal Mesh Simplification with Bounded Error*. IEEE Computer Graphics and Application vol.12 (3) (1996) pages 64-77

[9] O. Kreylos, B. Hamman. *On Simulated Annealing and the Construction of Linear Spline Approximations for Scattered Data*. IEEE Transactions on Visualization and Computer Graphics, vol. 7 (1) (2001) pages 17-31

[10] Mèrigot. *Revisiting Image Splitting*. In Proceedings of the 12th IEEE International Conference on Image Analysis and Processing (ICIAP'03), Mantova, Italy (2003) pages 314-319

[11] Quint. *Scalable Vector Graphics*. IEEE Multimedia vol.3 (2003) pages 99-101

[12] H. Samet. *Applications of Spatial Data Structures - Computer Graphics, Image Processing, and GIS*. Addison - Wesley Publishing Company, (1990) ISBN: 020150300X

[13] Scalable Vector Graphics (SVG) – XML Graphics for the Web – http://www.w3c.org/Graphics/SVG (2003)

[14] D. Su, P. Willis. *Demosaicing of Color Images Using Pixel Level Data-Dependent Triangulation*. In Proceedings of IEEE Theory and Practice of Computer Graphics (2003) pages 16-23

[15] Vector Eye – Raster to Vector Converter - http://www.siame.com/index.html (2003)

[16] X. Yu, B.S. Morse, T.W. Sederberg. *Image Reconstruction Using Data Dependent Triangulation*, Journal IEEE Computer Graphics And Applications, vol.21 (3) (2001) pages 62-68

[17] Colour Sensor Evaluation Kit VV6501, STMicroelectronics, Edinburgh, www.edb.st.com/products/image_sensors/501/6501evk.htm

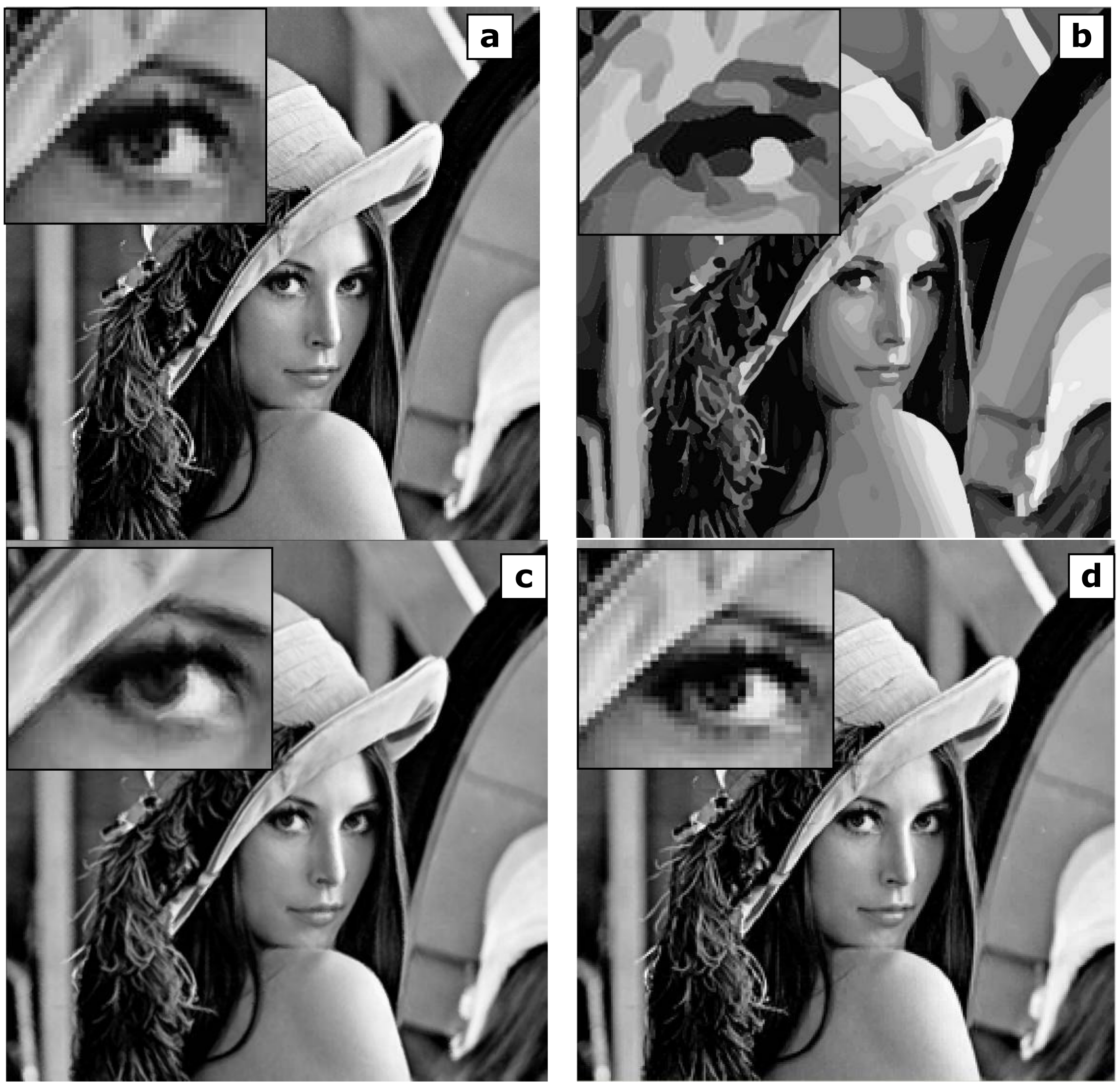


Figure 4. SVG rendering of standard "Lena" image (a), obtained by Vector-Eye software (b), proposed method (c) and quad-tree decomposition (d).

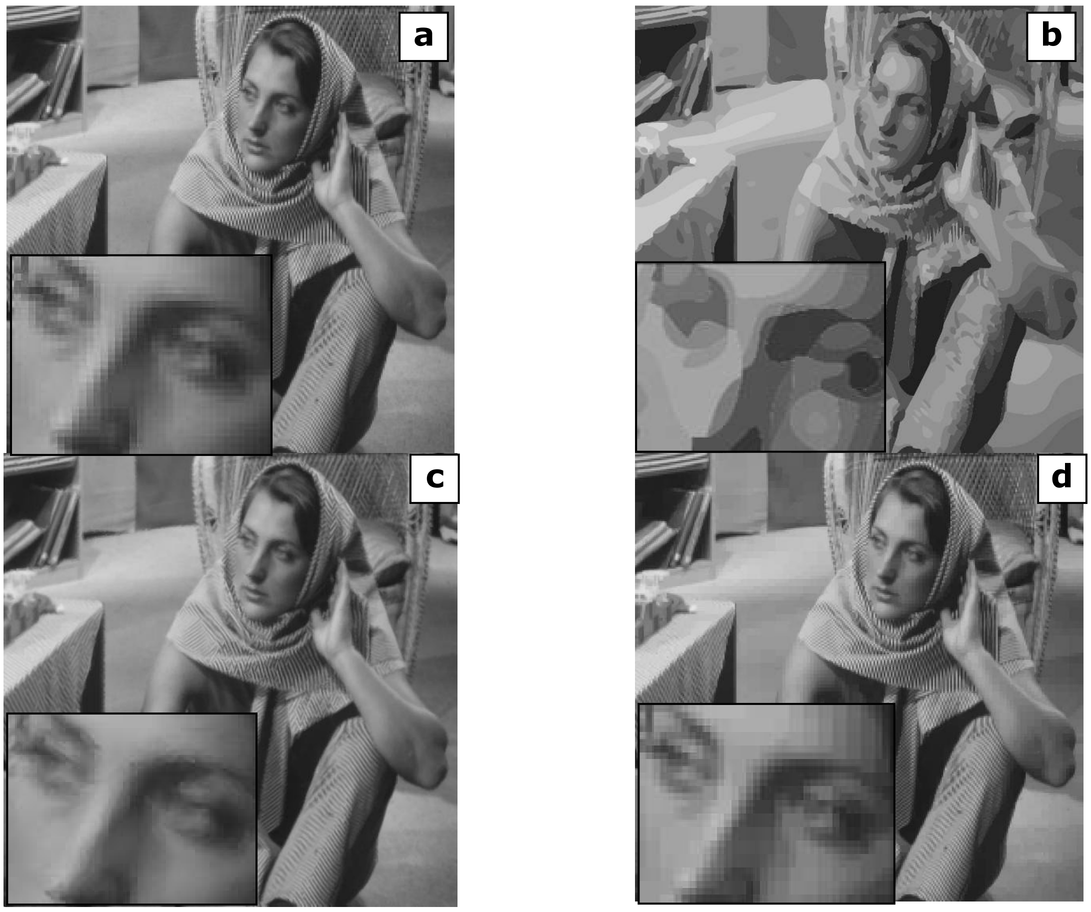


Figure 5. SVG rendering of standard "Barbara" image (a), obtained by Vector-Eye software (b), proposed method (c) and quad-tree decomposition (d).

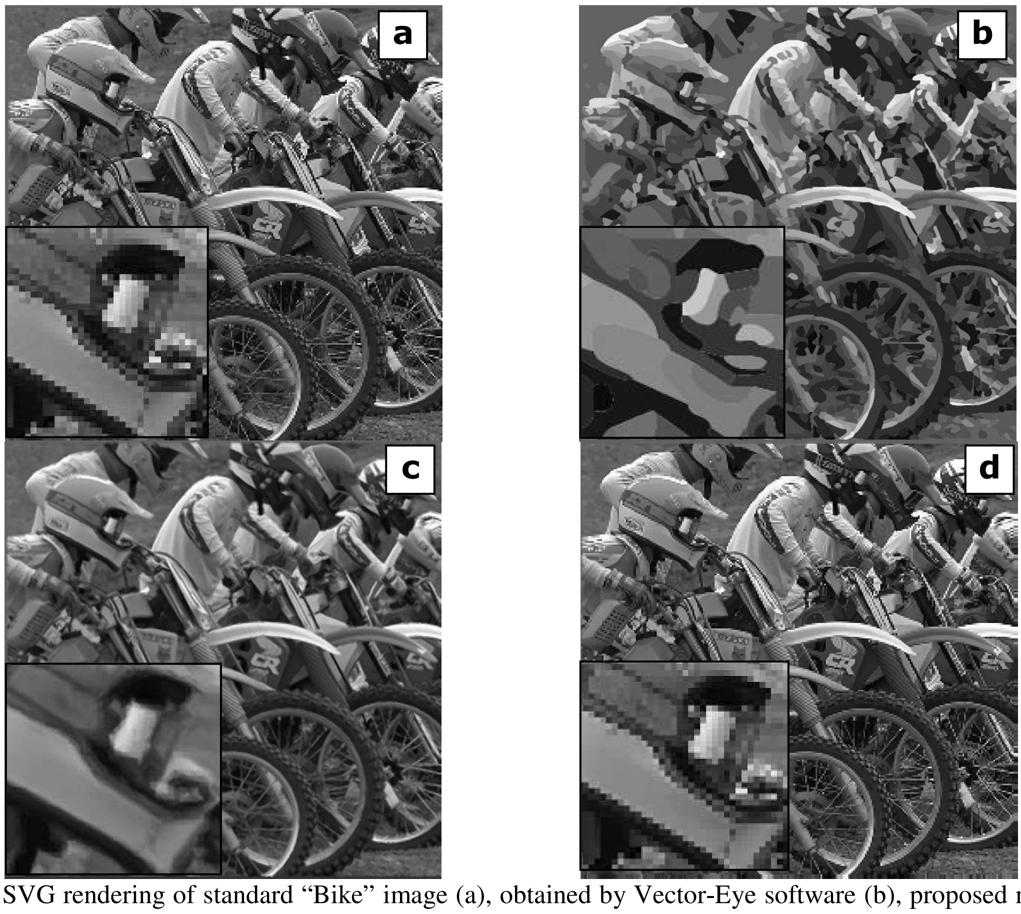
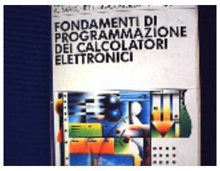
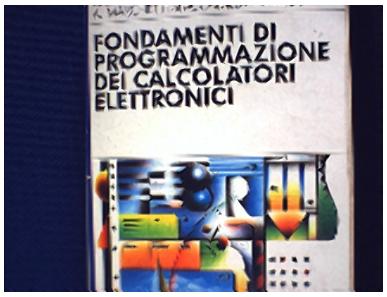


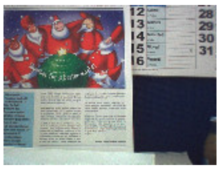
Figure 6. SVG rendering of standard "Bike" image (a), obtained by Vector-Eye software (b), proposed method (c) and quad-tree decomposition (d).



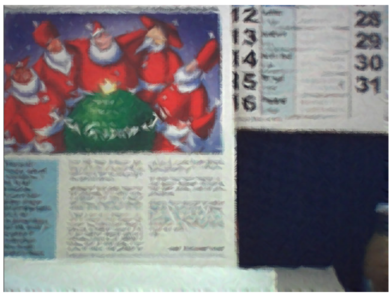
a.1



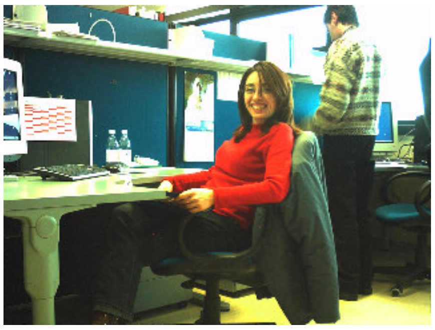
a.2



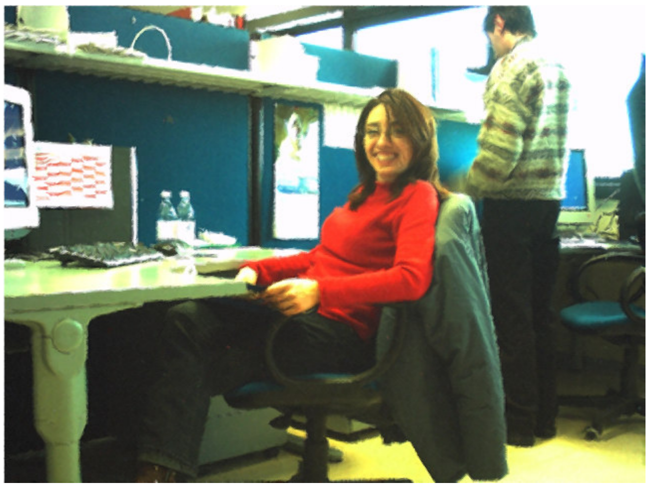
b.1



b.2



c.1



c.2

Figure 7. SVG rendering of images acquired by STV6500 CMOS sensor. The original raster images a.1) and b.1) are in QVGA format (160x120 pixels) while c.1) is in CGA format (320x240). All images have been rendered in SVG by the proposed technique, as shown in a.2), b.2), c.2), doubling original resolution.