Visual Comput (2005) 21: 373–383
DOI 10.1007/s00371-005-0292-4

ORIGINAL ARTICLE

Artificial mosaics

Gianpiero Di Blasi
Giovanni Gallo

Published online: 9 June 2005
© Springer-Verlag 2005

G. Di Blasi · G. Gallo
D.M.I., University of Catania,
Viale A. Doria, 6, 95125 Catania, Italy
e-mail: {gdiblasi,gallo}@dmi.unict.it

Abstract Art often provides valuable insight that can be applied to technological innovations, especially in the fields of image processing and computer graphics. In this paper we present a method to transform a raster input image into a good-quality mosaic: an "artificial mosaic." The creation of mosaics of artistic quality is challenging because the tiles that compose a mosaic, typically small polygons, must be packed tightly and yet must follow and emphasize orientations chosen by the artist. The proposed method can reproduce the colors of the original image and emphasize relevant boundaries by placing tiles along edge directions. No user intervention is needed to detect the boundaries: they are automatically detected using a simple but effective image processing technique. Several examples reported in the paper show how the right mixture of mathematical tools together with time-tested ideas of mosaicists may lead to impressive results.

Keywords Mosaic · Nonphotorealistic rendering · Distance transform · Image processing and enhancement

1 Introduction

Artists often invent techniques later used in computer graphics. Mosaics, for example, are images made by cementing together small colored tiles. They are probably the first example of image synthesis techniques based on discrete primitives.

A judicious use of orientation, shape, and size may allow one to convey much more information than the uniform distribution of N graphic primitives (like pixels, dots, etc.). For example, ancient mosaicists avoided lining up their tiles in rectangular grids because such grids emphasize only horizontal and vertical lines. Such artifacts may distract the observer from seeing the image that the tiles describe. Instead, old masters placed tiles in order to emphasize edges in the image, as can be seen in Fig. 1.

A review of existing mosaics shows that there are indeed several ancient "opera" that are nowadays called mosaics [8, 14]. Indeed we are interested in the most "pictorial" ones called "opera musiva," or "works worthy of the muses," i.e., of great visual refinery and effect.

An accurate and close observation of an opus musivum leads us to the following conclusions about "traditional looking" mosaics. They are made of tiles with the following properties:

1. Each tile has a uniform color;
2. Tiles may change in size and polygonal shape within some reasonable ranges and are generally convex;
3. "Empty" spaces between tiles should not only be reduced to a minimum but should be used as a graphical element to strengthen borders, lines, and edges.

In developing our artificial mosaic algorithm we assumed these properties as a reference, although the convexity constraint about tile shapes is somehow weakened.

Formalization into a computational framework of the mastery of artisans and of opera musiva is not easy: two basic tasks have to be modeled: global (or large scale)

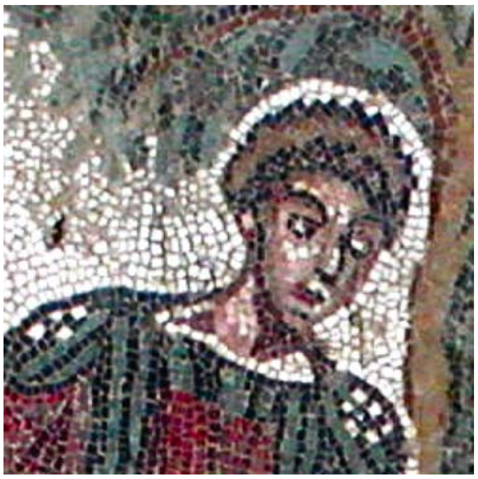


Fig. 1. Detail from "Great Hunting Scene," Villa del Casale, Piazza Armerina (Sicily), 4th century AD

optimization of the tile pattern and local (or small scale) optimization.

Globally, tiles have to agree with major feature curves. These curves provide the "directional guidelines" that characterize the semantic of the picture one wishes to render in mosaic. Directional guidelines are related to the salient edges of the image, and automatically providing such guidelines is in itself an interesting and challenging problem. Moreover, directional guidelines and edges are two related but different features. Thus the use of classical edge detector algorithms (see, for example, [13]) cannot be extrapolated without great care to the problem at hand.

Locally, tiles have to be packed in such a way as to minimize the interstices while at the same time not varying too much from some basic polygonal shape.

A good algorithm for automatically generating mosaics must address all of the above issues, and, since these issues are partially conflicting, the algorithm should also strike a good compromise between them.

1.1 Mosaics as an optimization problem

Hausner [7] and Elber and Wolberg [3] are the main authors who have tried to reformulate in a rigorous mathematical framework the problem of producing artificial mosaics. In particular, the mathematical formulation given by Hausner gives the mosaic construction from a source raster image in terms of a mathematical optimization problem as follows:

Given a rectangular region I^2 in the plane R^2, and a vector field $\Phi(x, y)$ defined in that region, find N sites $P_i(x_i, y_i)$ in I^2 and place N squares of side s, one at each P_i, with sides approximately parallel to $\Phi(x, y)$, such that all squares are disjoint and the area they cover maximized [7, p. 573].

The best aesthetic results are obtained when Φ aligns the tiles with the directional guidelines. Within this framework the problem can be viewed as a particular case of the "cover problem" or a "search and optimization problem."

The mosaic construction as formulated above can also be regarded as a "low-energy configuration of particles" problem, and the solution proposed in [7] can be extended to other problems in computer graphics and visualization, such as generating low-discrepancy sampling patterns and optimal vector field visualization.

Although Hausner's results are impressive, we believe that a cost function incorporating the many requirements of artistic mosaics should be much more complex than his proposal, and to the extent that it is, it would be hard to specify and minimize. We singled out at least five features that should be optimized; because these features are partially conflicting, the resulting global cost function is likely to be hard to minimize. In particular, the features that should be taken into account are:

1. The number of used tiles;
2. A measurement of the degree to which each tile orientation disagrees with $\Phi(x, y)$;
3. The percentage of the total image area covered by the tiles;
4. The percentage of surface where tiles overlap, in other words, the percentage of tiles that have to be "cut" from their original shape in order to fit in the mosaic;
5. The quality of the obtained mosaic
    - From a technical point of view (for example using the PSNR, peak signal-to-noise ratio, or analogous measurement);
    - From an aesthetic point of view (evaluating it algorithmically).

Some conflicts between the above features are evident. For example, when the tile size is fixed, if we try to minimize the number of tiles and/or to minimize the lost cut tiles, then probably the covered area will not be optimized; if we try to optimize the covered area, then the lost cut tiles or the angle orientation error could be very high; if we use, for aesthetic purposes, large amounts of grout space, then the covered area will be negatively affected; etc. Optimizing the cost function means striking a good balance among these conflicting properties, clearly not an easy task.

The optimization approach becomes even harder if one chooses to model directional guidelines with splines ([3]). This formulation has the advantage of providing mathematically sounded curves to place the mosaic tiles through offset spline computing. On the other hand, a solution that can deal in an aesthetically pleasing way with the cusps and other singularities that arise with this approach is not easy to formulate. This suggests that we should depart from a rigorous mathematical framework and adopt a purely image-space-based approach.

We propose a heuristically based, efficient, and direct geometric construction scheme that leads to realistic mosaics with a good visual impact. Surprisingly, our approach, even if it does not directly address the minimization of the proposed cost function, seems capable of performing an approximate optimization. This may suggest, if mathematical optimization of the cost function is a priority, using our mosaic as a suitable starting configuration for an optimization algorithm. Our geometric scheme is similar to but computationally simpler than [3] and starts from different heuristic principles. Moreover, the aesthetic results that we achieve are striking.

Finally, our algorithm for mosaic rendering is completely automated and requires no human input. In particular, there is no need to manually select the directional guidelines in the input image or at other stages of mosaic production.

The rest of this paper is organized as follows. In Sect. 2 we summarize previous related work. In Sect. 3 we explain our algorithm. In Sect. 4 we show the experimental results and compare our results with those obtained using other techniques. Finally, in Sect. 5 we suggest directions for future work and research.

2 Related work

Attempts in computer graphics to simulate mosaics fall into the broader area of nonphotorealistic rendering (NPR). In this section we limit our review to published works explicitly calling themselves "mosaics." Although mosaics are a traditional art form, attempts to simulate them in the digital realm are recent. Commercial image processing software (the examples in Fig. 2i,ii were produced using Adobe Photoshop) provide "mosaic filters" to obtain tessellated images whose appearance is far from the traditional mosaics.

More sophisticated approaches try to adopt smart strategies using computational geometry together with image processing. Haeberli [5] used Voronoi diagrams, placing the sites at random and filling each region with a color sampled from the image. This approach tessellates the image, but tile shapes are variable and do not follow edge features (Fig. 2iii). This technique is also available in many end-user applications, usually under the name "crystallization," and simulates the typical effect of stained-glass windows in churches.

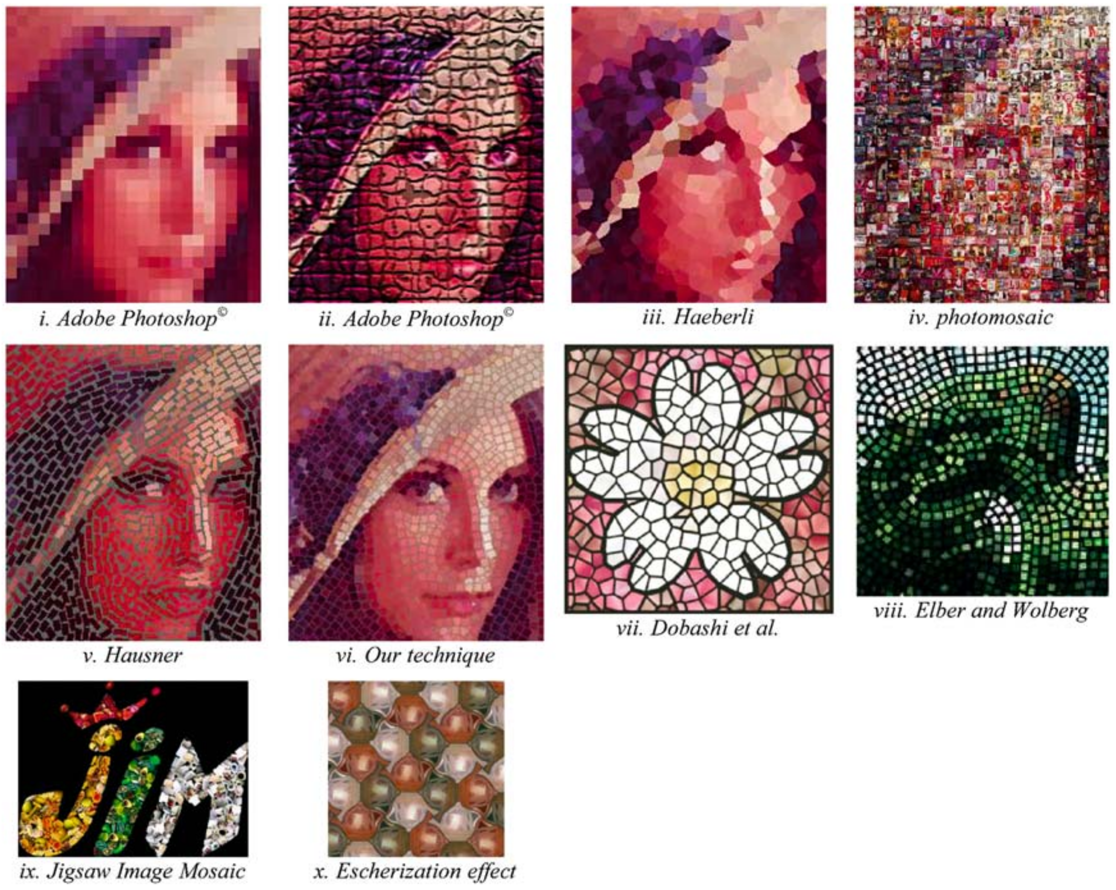


Fig. 2. Mosaic effects

"Photomosaic" [15] transforms an input image into a rectangular grid of thumbnail images (Fig. 2iv). In this approach the algorithm searches in a large database of images for one that approximates a block of pixels in the main image. The resulting effect is very impressive, but even in this case edge features are not respected and the effect is rather different than that of an ancient mosaic. The idea was later extended by Klein et al. [9] to videos to obtain a video mosaic.

Hausner [7] obtains very good results using centroidal Voronoi diagrams, edge features, L_1 (Manhattan) distance, and graphic hardware acceleration to optimize the results (Fig. 2v).

In [2] Dobashi et al. take up Haeberli's idea and obtain better results as they address the problem of taking into account the edges of the original image. Unfortunately, the tile shapes suffer from the extreme variability of Haeberli's technique (Fig. 2vii). Later on we directly compare our results with those of Dobashi et al. (Fig. 10).

Kim et al. [11] introduce a mosaicing technique where image tiles of arbitrary shapes are used to compose the final picture. The idea is quite similar to the photomosaic, but the final effect is very different and interesting (Fig. 2ix).

A very advanced approach to the rendering of traditional mosaics is presented in [3]. This technique is based on offset curves that get trimmed off the self, intersecting segments with the guidance of Voronoi diagrams. The algorithm requires a mathematical description, as B-splines, of the edges and allows very precise tile placement (Fig. 2viii). Another bonus of this approach is the use of variable-size tiles. Although the results are very good, the technique seems limited to cases of a single, user-selected, and close-edge curve.

For the sake of completeness, we also cite "Escherization" [10], a technique that produces tilings of the plane using a slightly distorted version of an image (Fig. 2x). It relies on symmetry groups and regular tilings. It is very different from the other methods reviewed above and is aimed at producing a sophisticated aesthetic effect unlike that of traditional mosaics.

3 The proposed algorithm

We begin this section with a more accurate discussion of what mosaicists do when they create a mosaic. Later we show how this may be translated into an algorithm.

3.1 How mosaicists work

To create a mosaic, the artisans first outline the shapes of the image they want to obtain, then they fill the shapes with a sequence of parallel (offset) curves, and, finally, they place the tiles along these curves. These concepts, illustrated in any standard "mosaic production" handbook (see, for example, [8]), are clearly depicted in Fig. 3.

The first two steps of mosaic creation are very simple and usually present no problems for mosaicists. The last step is the most complex one because mosaicists have a limited set of tile shapes. Usually only rectangular shapes are available, so mosaicists must adapt (by cutting) the tiles to insert them in the figure they are creating. This traditional approach to the problem, together with the commonly adopted solutions, is very clearly depicted in Fig. 3.

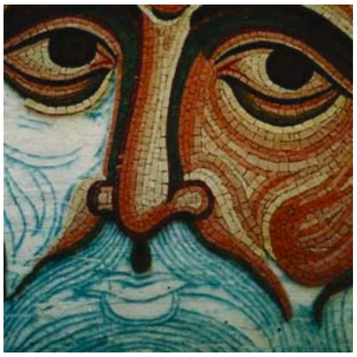


Fig. 3. How mosaicists work (image from [16])

3.2 Directional guideline detection

The first step of our algorithm is to find the directional guidelines of an image. In this section we will present a technique for automatically detecting guidelines. To solve this kind of problem, the edge detection algorithms available in the literature (see, for example, [13]) are of little use because here we are searching for "directional guidelines," which are perceptual features not always identifiable with conventional edges, especially in the case of photographic images (Fig. 4). Observe that what we call "directional guidelines" is closely related to Marr's primal sketch idea ([12])

Our technique for computing guidelines is very simple but effective. We work on the luminance channel of an image and start performing a histogram equalization. We then convolve the image with the origin-centered 2D Gaussian function ($\sigma = 16$):

$$G(x, y) = \frac{1}{2\pi\sigma^2} \mathrm{e}^{-\frac{x^2+y^2}{\sigma^2}} .$$

This leads to a new image I_1 (a smoothed version of the original image). Let $\boldsymbol{\mu}$ be the mean value of I_1, and let $\boldsymbol{\Sigma}$ be the variance value. Hence it is possible to compute the I_2 image given by the function:

$$I_2(x, y) = \begin{cases} 1 & \textit{if } |I_1(x, y) - \boldsymbol{\mu}| > T , \quad \textit{where } T = \frac{\boldsymbol{\Sigma}}{4} , \\ 0 & \textit{elsewhere.} \end{cases}$$

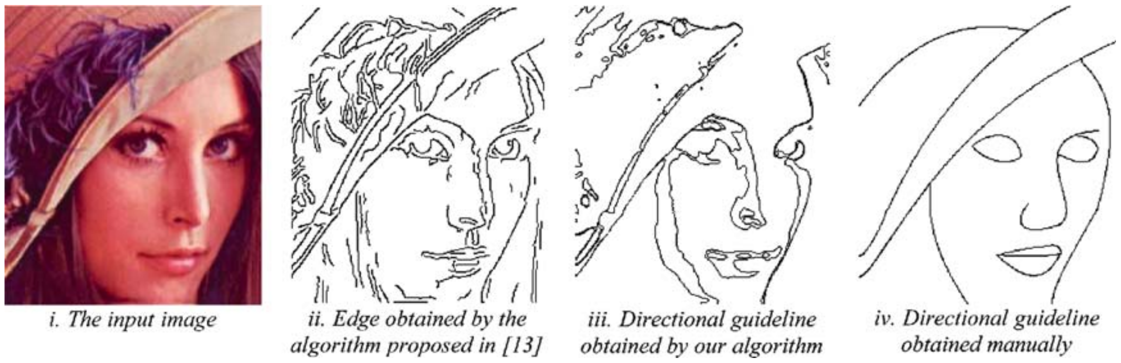

i. The input image
ii. Edge obtained by the algorithm proposed in [13]
iii. Directional guideline obtained by our algorithm
iv. Directional guideline obtained manually

Fig. 4. Edge detection vs. directional guideline detection

The threshold value T has been experimentally selected, through trial and error, to provide the best results in the successive processing. Finally, we convolve I_2 with a Laplace edge detector obtaining, after removing isolated points, the directional guidelines. Figure 5 illustrates the successive steps of the algorithm. We tested this technique on a large set of images and obtained good results with paintings, graphics, or photos as source images.

The problem of defining and finding directional guidelines is an interesting computer vision problem and is beyond the scope of this paper. Although we have adopted here a practical solution that is satisfactory for the application at hand, the problem deserves deeper theoretical and algorithmic investigation.

3.3 How we copied the mosaicists' work

Suppose we have an image and its directional guidelines as input (Fig. Fig. 6i,ii). Using the directional guidelines we evaluate for each pixel of the image the distance transform [6], i.e., its minimum distance from any guideline pixel, obtaining the matrix (*dtM*) illustrated in Fig. 6iii (here nearest pixels are white, farthest pixels are black, and guideline pixels are yellow). The use of the distance transform in the field of NPR was previously proposed, for a different purpose, by Gooch et al. [4].

Starting from the distance transform matrix we obtain another two matrices needed to perform the final mosaicing: the gradient matrix (*gM*) and the level line matrix (*llM*). These matrices are computed as follows:

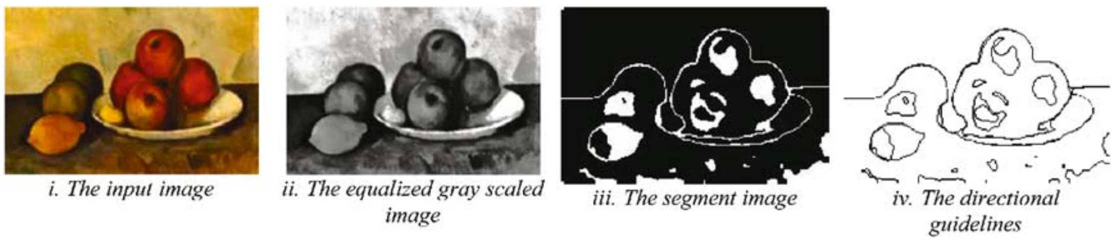

i. The input image
ii. The equalized gray scaled image
iii. The segment image
iv. The directional guidelines

Fig. 5. Directional guidelines detection

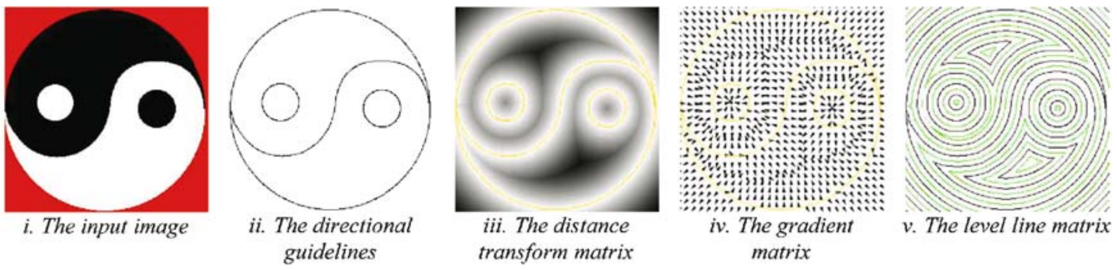


Fig. 6. Input of our algorithm and matrices used by the algorithm

$$gM(x,y) = \arctan \frac{dtM(x,y+1) - dtM(x,y-1)}{dtM(x+1,y) - dtM(x-1,y)}\,,$$

$$llM(x,y) = \begin{cases} 1 & \textit{if module}\ \ (dtM(x,y), 2\cdot tSize) = 0\,, \\ 2 & \textit{if module}\ \ (dtM(x,y), 2\cdot tSize) = tSize\,, \\ 0 & \textit{elsewhere}\,, \end{cases}$$

where *tSize* is a user-selected integer value that denotes the size of the tiles. Their meaning is clearly shown in Fig. 6iv,v (in Fig. 6v, black pixels have value 1, green pixels have value 2).

Observe that:

1. The tiles need not be square; they may have any aspect ratio. However, only one dimension, *tSize*, is required to compute the *llM* matrix.
2. The function used to compute the *llM* matrix can be easily adapted to prepare the image for accommodating various-size tiles as in [3].
3. Since *tSize* is an integer and matrix *dtM* has only integer entries, the module computation for matrix *llM* is always well defined.

We are now ready to place the tiles, initially all of the same shape and size, using the pixels in *llM* with value 2. Observe that such pixels form chainlike sequences. Of course, in the process of placing the tiles their shape has to be altered to resolve overlapping. More precisely, the algorithm proceeds as follows:

- While there are chains of pixels with value 2 not yet processed:
  a. Select a chain;
  b. Starting from an arbitrary pixel on it "follow" the chain;
  c. Place new tiles at regular intervals along the path (the orientation of the tiles is assigned using the gradient information from matrix *gM).*

The distance along the chain that separates successive tiles is equal to *sSize* when tiles of dimension *tSize* × *sSize* have been adopted.

If tiles of fixed size and shape are positioned only according to the method described, two main difficulties arise:

1. Tiles may overlap.
2. A single tile may cover an area across the "black pixel lines" (i.e., the pixels with value 1 in *llM*).

Both of these effects are unpleasant. In particular, the second problem completely destroys the guideline patterns and would result in blurred images.

To address these difficulties, we adopt a very simple strategy:

1. The overlapping of tiles is easily detected maintaining a boolean mask of covered pixels. If a tile that we are trying to place contains pixels already covered by previously placed tiles, then we change the original rectangular shape of the tile, "cutting away" the overlapping pixels (Fig. 7i,ii);
2. If a new tile crosses any "black pixel line," it is trimmed against this line (Fig. 7iii,iv).

Note that until now our tiles have been placed without any grout space.

Once the tile positioning and cutting phase is complete, two postprocessing steps must be performed to achieve a pleasant aesthetic effect.

First, as was pointed out earlier, "grout spaces" between tiles are important. To achieve the effect of cement showing through tiles, a downscaling of each tile is performed. This frees some pixels that will be assigned a unique color for concrete under the mosaic.

Second, we choose for each tile a uniform color equal to the color of the pixel corresponding to its center in the source image. Other choices may lead to different artistic effects. Some typical results of the reported mosaic technique are shown in Fig. 8.

Observe that in Fig. 8iv region A is what is left from a regular tile after it has been cut according rules 1 and 2 above; it is not the underlying image showing through the other tiles, as may be wrongly perceived at first glance.

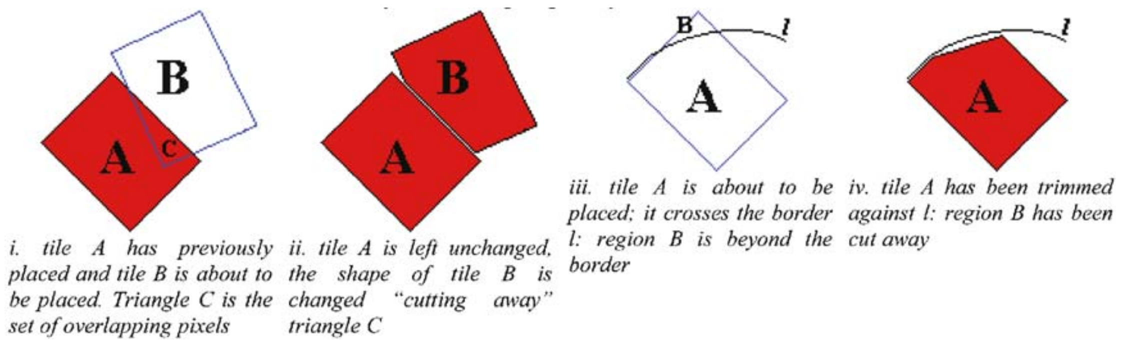

B
A
C
B
A
B
l
A
l
A
i. tile A has previously placed and tile B is about to be placed. Triangle C is the set of overlapping pixels
ii. tile A is left unchanged, the shape of tile B is changed "cutting away" triangle C
iii. tile A is about to be placed; it crosses the border l: region B is beyond the border
iv. tile A has been trimmed against l: region B has been cut away

Fig. 7. Cutting tiles

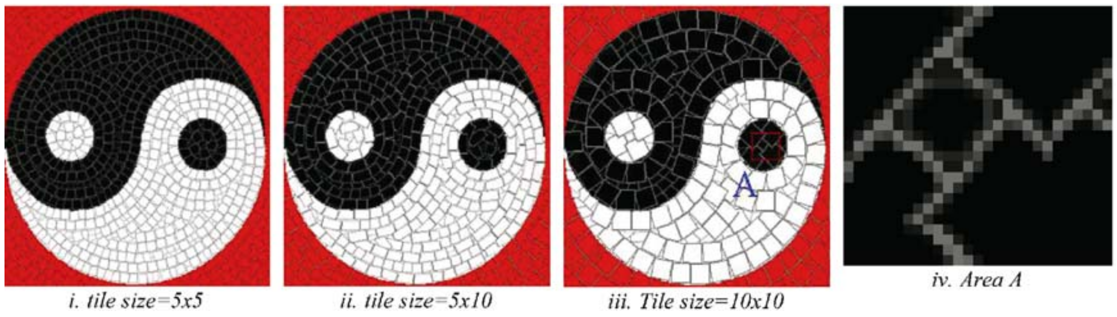


Fig. 8. Some "artificial mosaics" of the input image; in these cases the scale of the tiles is smaller than the details in the source image

3.4 Enhancements

In this section we point out some limitations of our method and propose possible solutions. The limitations can be easily explained observing Fig. 9i. The figure shows that when the tile size achieves a scale greater than the details in the source picture, some of them have a badly nonconvex shape or a curved boundary. This is not observed in real mosaics and hence is not desirable in our "artificial" ones.

An obvious "cure" to the problem is to keep the dimensions of the tiles small (typically, below 5% of the image dimension). Note that the scale problem (first addressed in [3]) is not a problem of our algorithm but rather an intrinsic problem of mosaic art and production: a wrong choice of the tile scale will result in a bad final effect, independently of the adopted algorithm.

To address these problems, we implemented some simple heuristic techniques to remove the arcs and concavities from a shape. One of the rules is to check with simple contour tracking the presence of tiles with curved (not straight) boundaries. This first rule has been, for example, applied in area A of Fig. 9i, magnified in Fig. 9iii. The other rule aims at eliminating concave boundaries. Whenever a concavity is detected, the tile is split into two smaller convex tiles. This rule has been applied, for example, in area B of Fig. 9i, magnified in Fig. 9v. Using these heuristics the mosaic of Fig. 9i can be converted into the mosaic shown in Fig. 9ii. Figure 9iv,vi shows areas A and B after application of the "cure."

To obtain even better realism, other "tricks" are implemented. For example, if during processing a tile is cut into a too small region, it is advisable to simply discard all of it. This simple rule imparts to the mosaic a ruined and aged appearance, which is very interesting for some image classes.

4 Experimental results and examples

To illustrate the effectiveness of the proposed technique, we report some examples and quantitative results. The algorithm was implemented in Java2 Standard Edition 1.4.2, and all experiments were carried out on an Athlon XP-M 1800+ PC, 192 MB RAM, using Windows XP Home Edition. The reader may directly test the quality of our algorithm by downloading the Artificial Mosaic Creator applet at `http://www.dmi.unict.it/`

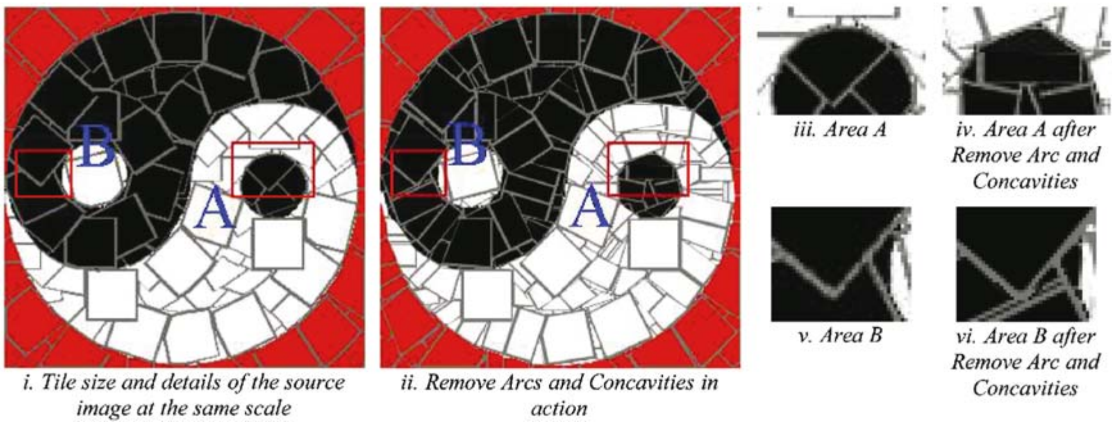

B
A
i. Tile size and details of the source image at the same scale
B
A
ii. Remove Arcs and Concavities in action
iii. Area A
iv. Area A after Remove Arc and Concavities
v. Area B
vi. Area B after Remove Arc and Concavities

Fig. 9. Another "artificial mosaic" obtained from Fig. 6i

~gdiblasi/mosaic/mosaic.html , and by downloading a JGimp plug-in and a Java application at http://www.dmi.unict.it/~gdiblasi/mosaic/mosaic.jar. Our implementation at this time does not rely on any particular graphic hardware acceleration.

We compare our performance with Hausner's method in order to assess the quality of our artificial mosaics. Note that the comparison is not completely fair because, as was pointed out, the two algorithms have slightly different objectives. Direct comparison with the technique of [3] has not been carried out because, although the mathematical foundations of that algorithm are deep, the published results are not fully conclusive from an aesthetic point of view.

A mosaic made up of N tiles conveys more information than an image made up of N rectangular pixels in a regular lattice. We can see this by comparing Fig. 2i with Fig. 2vi, which better conveys the directional guidelines. Other examples are shown in Figs. 10–12. Figure 12, in particular, illustrates the likeness between our "artificial mosaic" and Byzantine masterpieces. We also found very appealing the results obtained using modern art works like Cezanne's paintings (last image in Fig. 10 and Fig. 11). The first image in Fig. 10 proves the effectiveness of the proposed technique when the source image is a good-quality photograph. Moreover, our results compare well with analogous results by [2] (Figs. 2vii and 10iii).

Timing results (Table 1) show that our algorithm is fast enough for use as a plug-in in typical end-user software. It is straightforward to verify that the time asymptotic complexity is linear in the number of pixels of the image.

To compare our method with Hausner's method, let us recall the features that should be taken into account in a cost function incorporating the requirements of artistic mosaics:

1. The number of used tiles (*nT*, number of tiles);
2. A measurement of the degree to which each tile orientation disagrees with $\Phi(x, y)$ (*vfd*, vector field deviation);
3. The percentage of the total image area covered by the tiles (*cA*, covered area);
4. The percentage of surface where different tiles overlap (*lct*, lost cut tiles);
5. The quality of the obtained mosaic (from technical and aesthetic points of view).

Using Hausner's method, *vfd* and *lct* will always be minimized, *nT* will be generally small, but *cA* could also turn out too small. Our method allows the tuning of several parameters in order to obtain a good compromise; for example, we can easily obtain the maximum value for *cA*, but at the price of a very high value for *lct* and *nT*. To confirm our qualitative observations, we selected two reference images: the "Yin and Yang" image and the Lena image. For comparison we used Hausner's results, which may be downloaded from the Web. We created the mosaics using our method with tiles as close in size as possible to Hausner's tiles, and we compared the two images. Results are summarized in Table 2.

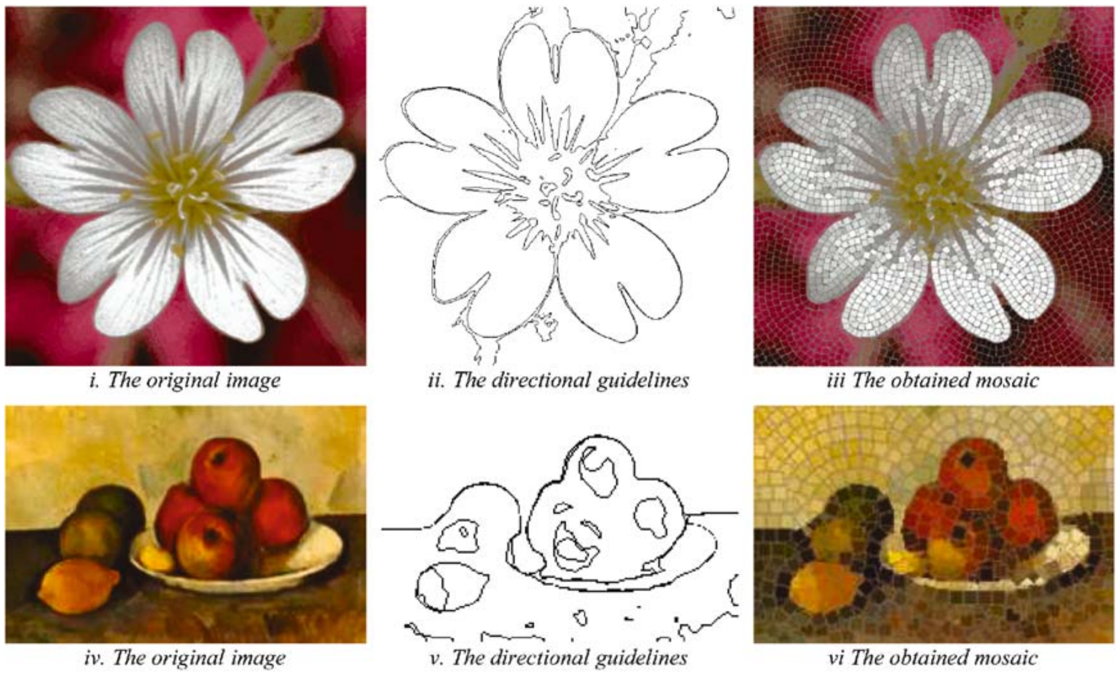
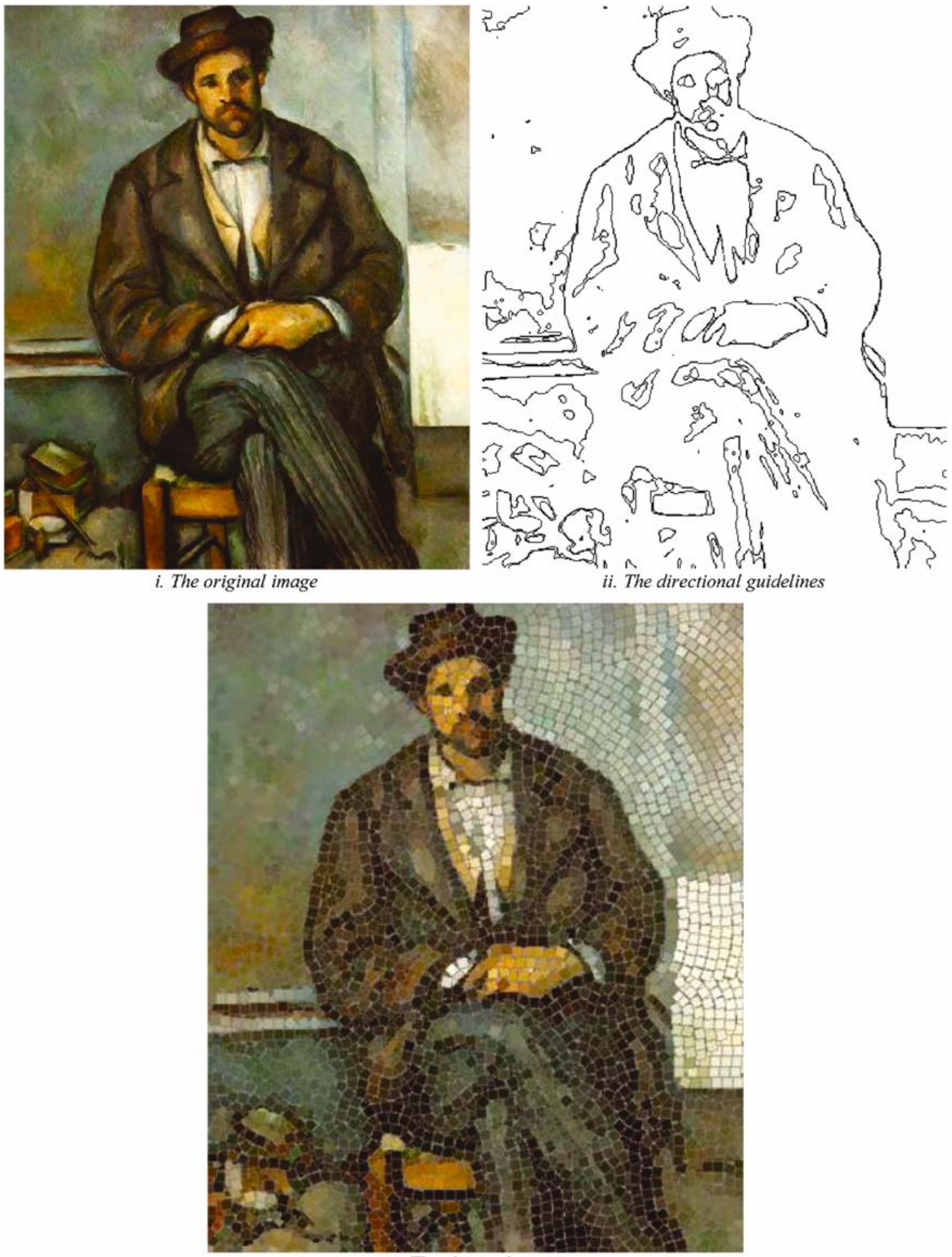


Fig. 10. Some examples



i. The original image *ii. The directional guidelines*

iii. The obtained mosaic

Fig. 11. The mosaic version of a Cezanne painting

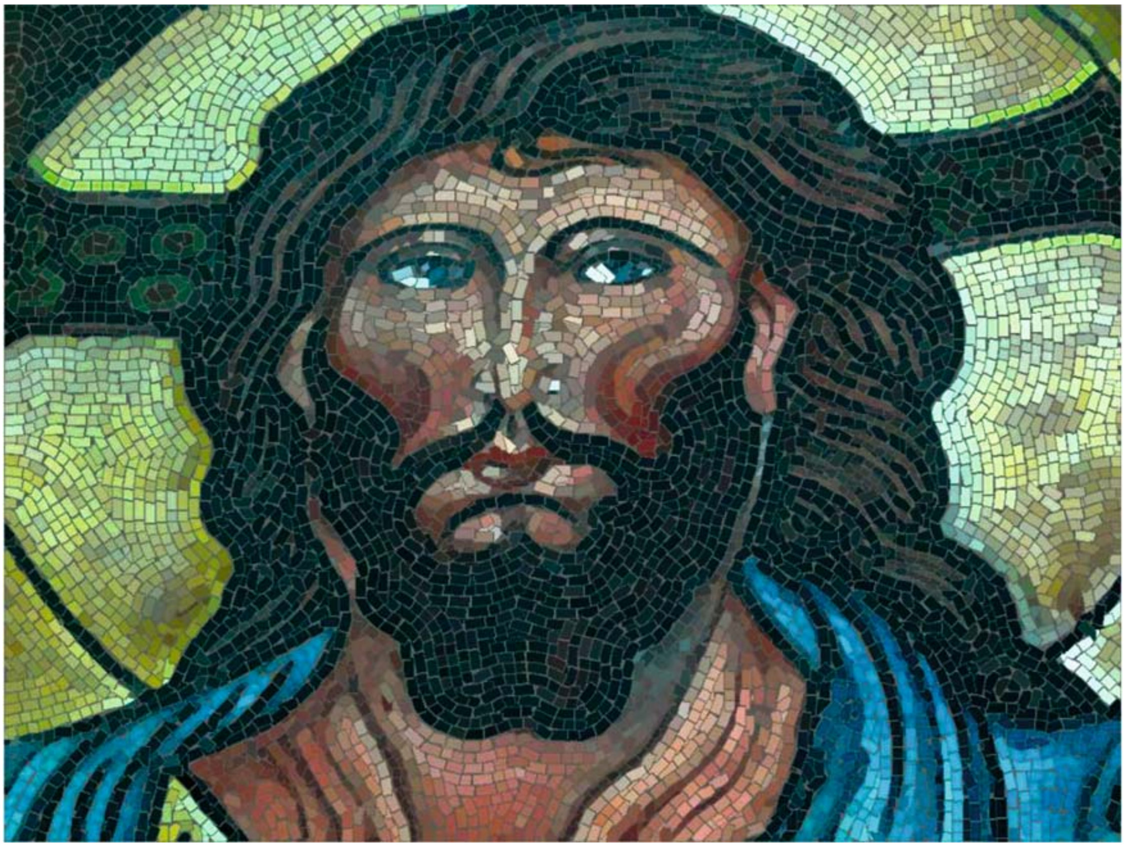


Fig. 12. The Pantocrator mosaic

Table 1. Timing results

Size	Guideline mean detection time (s)	Mosaic mean time (s)	Total mean time (s)
600×600	1.402	10.485	11.887
800×600	1.402	12.689	14.091
593×886	1.602	15.182	16.784
1024×768	2.243	22.142	24.385

An aesthetic comparison between the two techniques can forcibly be only qualitative. We selected two aesthetic features:

1. Uniform tile visual distribution and
2. Overall visual appeal.

Objective measurement of these features is of course very hard for technical (nonartist) experimenters. We attempted to obtain an objective measurement by asking several art students to express their subjective preference for pairs of mosaics obtained with the two different techniques in a kind of blind test relative to the two features reported above.

Regarding visual uniformity, Hausner's mosaics performed slightly better than ours. When overall visual appeal was at issue, our mosaics were by and large preferred.

Table 2. Comparing results

Image	Hausner's method				Our method			
	vfd	nT	cA	lct	vfd	nT	cA	lct
Yin and yang	0	500	77%	∼ 0%	0	712	92%	43%
Lena	0	4000	65%	∼ 0%	0	4943	92%	52%

5 Conclusions & future work

In this paper we presented a new method to create artificial mosaics, achieving an impressive visual impact. Experimental results show the soundness of our algorithm.

The aesthetic of our results several ideas that were described for the first time in this work could be improved upon:

1. A different strategy for choosing, in the case of overlap, the tile to be cut. Heuristic rules or, perhaps, randomized choices could produce different outcomes.
2. Automatic optimized choices of tile scale relative to each input image is an open problem worthy of further investigation.
3. Generalization of our "mosaicist heuristic" to other kinds of primitive-based nonphotorealistic image processing seems possible and quite promising.
4. Some generalizations proposed in [3], such as variable-size tiles and photomosaic, are also considered for future work and research. It would also be interesting to explore the possibilities offered using basic shapes besides rectangular tiles.
5. A different method for better locating directional guidelines is an important topic of further research.
6. Exploitation of hardware graphic primitives to accelerate the mosaic synthesis is also of considerable interest.
7. Extension of our method for mosaic rendering of 3D surface is probably the most exciting direction of research.

Acknowledgement We thank the anonymous referees, Giuseppe Patanè, Ettore Luzio, and Francesco Taranto for their precious suggestions and the technical support that they provided us during this research.

References

1. Di Blasi G, Gallo G (2003) The Artificial Mosaic Creator applet www.dmi.unict.it/~gdiblasi/mosaic/mosaic.html, JGimp plug-in and Java application www.dmi.unict.it/~gdiblasi/mosaic/mosaic.jar
2. Dobashi J, Haga T, Johan H, Nishita T (2002) A method for creating mosaic images using Voronoi diagrams. In: Proceedings of Eurographics
3. Elber E, Wolberg G (2003) Rendering traditional mosaics. Visual Comput 19(1):67–78
4. Gooch B, Coombe G, Shirley P (2002) Artistic vision: painterly rendering using computer vision techniques. In: Proceedings of NPAR, pp 83–90
5. Haeberli P (1990) Paint by numbers. In: Proceedings of SIGGRAPH, pp 207–214
6. Haralick R, Shapiro L (1992) Computer and robot vision – vol 1. Addison-Wesley, Reading, MA
7. Hausner A (2001) Simulating decorative mosaics. In: Proceedings of SIGGRAPH, pp 573–580
8. King S (2003) Mosaic techniques and traditions: projects and designs from around the world. Sterling, New Delhi
9. Klein A W, Grant T, Finkelstein A, Cohen M F (2002) Video mosaics. In: Proceedings of NPAR, pp 21–28
10. Kaplan C, Salesin D (2000) Escherization. In: Proceedings of SIGGRAPH, pp 499–510
11. Kim J, Pellacini F (2002) Jigsaw image mosaics. In: Proceedings of SIGGRAPH, pp 657–664
12. Marr D (1982) Vision. Freeman, New York
13. Meer P, Georgescu B (2001) Edge detection with embedded confidence. Trans Patt Anal Mach Intell 23(12):1351–1365
14. Nittolo F (2004) Il mosaico. http://www.ravennarte.it/rarteing/mosaico.htm
15. Silvers R, Hawley M (1997) Photomosaics. Holt, New York
16. Tumminello S (2003) Descrizione e tecnica utilizzata nei mosaici del Duomo di Monreale. http://www.parrocchie.it/monreale/sscrocifisso/italia/mosaici.htm

GIANPIERO DI BLASI got his Laurea at Università di Palermo in 1999, Italy and he is candidate to receive a Ph.D. at Università di Catania in 2006. His research interests include Computer Graphics, non-photorealistic rendering and Java programming.

GIOVANNI GALLO got his Laurea at Università di Catania, Italy and his Ph.D. at New York University in 1992. Since then his research interests include images, graphics and visualization. He currently is Full Professor of Computer Science at Università di Catania.