DOI: 10.1111/j.1467-8659.2007.01021.x

COMPUTER GRAPHICS *forum*
Volume 26 (2007), number 4 pp. 794–812

Digital Mosaic Frameworks - An Overview

S. Battiato, G. Di Blasi, G. M. Farinella and G. Gallo

Dipartimento di Matematica ed Informatica, Università di Cataniavia,
A. Doria, 6 – 95125 Catania, Italy
email:{battiato, gdiblasi, gfarinella, gallo}@dmi.unict.it

Abstract
Art often provides valuable hints for technological innovations especially in the field of Image Processing and Computer Graphics. In this paper we survey in a unified framework several methods to transform raster input images into good quality mosaics. For each of the major different approaches in literature the paper reports a short description and a discussion of the most relevant issues. To complete the survey comparisons among the different techniques both in terms of visual quality and computational complexity are provided.

Keywords: Artificial Mosaic, Non-Photo-realistic Rendering

ACM CCS: J.5 Arts and Humanities: *Architecture, Fine Arts*, I.3.3 Computer Graphics: *Picture/Image Generation.*

1. Introduction

Nonphotorealistic Rendering (NPR) is a successful area of Computer Graphics and it is nowadays applied to many relevant contexts: Scientific Visualization, Information Visualization and artistic style emulation [ST90,Col04]. NPR's goals may be considered complementary to the traditional main goal of the Computer Graphics which is to model and render 3D scenes in a natural (i.e. photorealistic) way.

Within NPR the recent approach to digitally reproduce artistic media (such as watercolors, crayons, charcoal, etc.) and artistic styles (such as cubism, impressionism, pointillism, etc.) have been gaining momentum and are very promising [CH03,CAS*97 and HJO*01]. Several denominations have been proposed for this narrower area within NPR: Artistic Rendering (AR) [Col04] and Computational Aesthetics (CA) [Com06] are among the most popular ones. None of these names have reached general acceptance within the research community. The authors of this paper believe that, beyond names, a proper definition of the area that explains in a suitable way its purpose and its aim may be the following: *to reproduce the aesthetic essence of arts by mean of computational tools*.

The focus of this paper is to review the state of the art for the problem of digital mosaic creation. The survey restricts its scope only to the techniques that explicitly bear the 'mosaic' name and that make use of primitives larger than pixels, points or lines. Stippling [DHVS00,Sec02,HHD03,SGS05] and hatching [SHS02,SGS05] are hence not covered here, although their visual similarity to mosaic naturally leads to approaches for these techniques that closely resemble the mosaic techniques.

Mosaics, in essence, are images obtained cementing together small colored fragments. Likely, they are the most ancient examples of discrete primitive based images. In the digital realm, mosaics are illustrations composed by a collection of small images called 'tiles'. The tiles tessellate a source image with the purpose of reproducing the original visual information rendered into a new mosaic-like style. The same source image may be translated into many strikingly different mosaics. Factors like tile dataset, constraints on positioning, deformations and rotations of the tiles are indeed very influent upon the final results. As an example, the creation of a digital mosaic resembling the visual style of an ancient looking man-made mosaic is a challenging problem because it has to take into account the polygonal shape of the tiles, the small size of the tiles, the need to pack the tiles as densely as possible and, not last, the strong visual influence that tile orientation has on the overall perception of themosaic. In particular orientation cannot be arbitrary but it is constrained to

Submitted May 2006
Revised December 2006
Accepted January 2007

follow the gestalt choices made by the author of the source picture. Tiles, hence, must follow and emphasize the main orientations chosen by the artist.

A first step toward the solution of the problem of digital mosaic creation is to state it within a mathematical framework. In particular the translation of a raster source image into a digital mosaic may take the form of a mathematical optimization problem as follows:

Given a rectangular region I^2 in the plane R^2, a tile dataset and a set of constraints, find N sites $P_i(x_i, y_i)$ in I^2 and place N tiles, one at each P_i, such that all tiles are disjoint, the area they cover is maximized and the constraints are verified as much as possible.

The above definition is general and is suitable for many applications beyond Computer Graphics field. Indeed Harmon in 1973 [Har73] published the first results related with this kind of problems in the context of modeling human perception and automatic pattern recognition (see below). Within this framework the problem can be viewed as a particular case of the cover problem or as a search and optimization problem. The mosaic construction as formulated above can also be regarded as a low-energy configuration of particles problem.

In the specific case of mosaics four different definitions can be given to solve specific problems:

Crystallization Mosaic *- Given an image I^2 in the plane R^2 and a set of constraints (i.e. on edge features), find N sites $P_i(x_i, y_i)$ in I^2 and place N tiles, one at each P_i, such that all tiles are disjoint, the area they cover is maximized, each tile is colored by a color which reproduces the image portion covered by the tile. In this case in order to allow a solution the requirements have to be relaxed asking only that the constraints are verified as much as possible.*

Ancient Mosaic *- Given an image I^2 in the plane R^2 and a vector field $\Phi(x,y)$ defined on that region by the influence of the edges of I^2, find N sites $P_i(x_i, y_i)$ in I^2 and place N rectangles, one at each P_i, oriented with sides parallel to $\Phi(x_i, y_i)$, such that all rectangles are disjoint, the area they cover is maximized and each tile is colored by a color which reproduces the image portion covered by the tile [Hau01].*

Photo-mosaic *- Given an image I^2 in the plane R^2, a dataset of small rectangular images and a regular rectangular grid of N cells, find N tile images in the dataset and place them in the grid such that each cell is covered by a tile that 'resembles' the image portion covered by the tile.*

Puzzle Image Mosaic *- Given an image I^2 in the plane R^2, a dataset of small irregular images and an irregular grid of N cells, find N tile images in the dataset and place them in the grid such that the tiles are disjoint and each cell is covered by a tile that 'resembles' the image portion covered by the tile.*

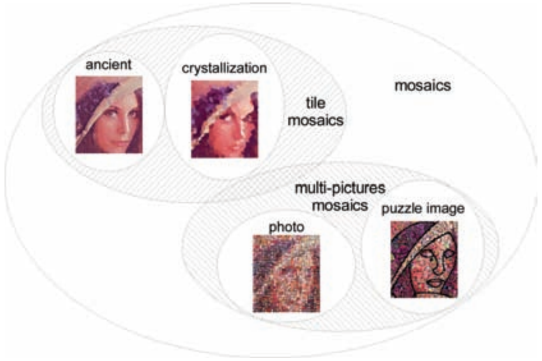


Figure 1: *Mosaic classification.*

Different solutions to the problems above have been proposed. Most of these solutions are reviewed in this paper within a unified framework. More precisely we single out four different mosaic types:

1. crystallization mosaics (a.k.a. tessellation);
2. ancient mosaics;
3. photo-mosaics and
4. puzzle image mosaics.

The first two types of mosaics decompose a source image into tiles (with different color, size and rotation), reconstructing the image by properly painting the tiles. They may hence be grouped together under the denomination of *tile mosaics*. The last two kind of mosaics are obtained by fitting images from a database to cover an assigned source image. They may hence be grouped together under the denomination of *multi-picture mosaics*. The proposed taxonomy should not be intended as a rigid one. Many mosaic techniques may fit in more than a single class and it is likely that other new types of mosaics will appear in the future (see Figure 1). Other classifications of the known techniques for digital mosaics may be chosen if other criteria are taken into account. For example mosaics could be classified into:

1. fixed tile and variable tile (picture) size;
2. Voronoi based and non-Voronoi based approach;
3. deterministic and nondeterministic (probabilistic, random) algorithm;
4. iterative and one-step method and
5. interactive and batch system.

Finally, the relative performances with respect to the computational complexity has to be taken into account for an effective evaluation and classification.

In the following most of all previously published algorithms are reviewed. The reader is provided with a general idea about the working of each technique. A discussion of the weakness and the strong points of each method is provided and, whenever possible, the relationships between algorithmic choices and visual qualities of the resulting images are emphasized.

The rest of this paper is organized as follows: in Section 2 we present the crystallization mosaic techniques, Section 3 explains the ancient mosaic methods. In Section 4 we review the photo-mosaic algorithms and in Section 5 the puzzle image mosaics are presented. Finally Section 6 and 7 are devoted to final discussions and suggestions for future work.

2. Crystallization Mosaics

Many sophisticated mosaic approaches adopt smart strategies using computational geometry (e.g. Voronoi diagrams) together with image processing. These techniques generally lead to mosaics that simulate the typical effect of some glass windows in the churches.

A Voronoi diagram is a geometric structure that represents proximity information about a set of points or objects [PS87]. Given a set of sites or objects, the plane is partitioned by assigning to each point its nearest site. The points whose nearest site is not unique, form the Voronoi diagram. That is, the points on the Voronoi are equidistant to two or more sites. From a geometric point of view Voronoi cells can be considered convex polygons. Voronoi diagrams were first discussed by Lejeune-Dirichlet in 1850, but it was more than a half of a century later, in 1908, that these diagrams were written about in a paper by Voronoi, hence the name Voronoi Diagrams. The Voronoi cells/polygons are sometimes also called Dirichlet Regions. There are a variety of algorithms available to construct Voronoi diagrams (see for example [PS87 and DVOS97]), but the most famous algorithm was presented by Fortune [For87]; he developed a plane-sweep algorithm which is more efficient in time than any other incremental algorithm. The algorithm guarantees an $O(nlg(n))$ complexity in the worst case.

Haeberli [Hae90] used Voronoi diagrams, placing the sites at random and filling each region with a color sampled from the image. This approach tessellates the image with tiles of variable shapes and it does not attempt to follow edge features; the result is a pattern of color having a cellular-like look (see Figure 2b). The effect may be efficiently implemented by z-buffering a group of colored cones onto a canvas. This allows to make the best use of hardware acceleration provided by modern graphics cards. Although very simple, Haeberli's idea is a milestone in this field and a starting point for many subsequent techniques.

In [DHJN02] Dobashi *et al.* extended the original idea of Haeberli. Their results are aesthetically more pleasant because the technique that they propose integrate edge information with Voronoi tessellation. The method, however, suffers from the variability that Haeberli's algorithm produces on tile shapes (Figure 2c). The strategy to better approximate the source image is simple. An error function E for the output image is defined as:

$$E = \sum_{x,y,c} \left(P^{Input}_{(x,y,c)} - P^{Output}_{(x,y,c)} \right)^2 \tag{1}$$

The function E is minimized by iteratively moving the centers of the Voronoi's polygons; the movement is limited to the 8-pixel neighborhood. The authors introduce also an heuristic strategy to speed up the moving process.

Recently, Faustino and Figueiredo [FF05] presented a technique similar to Dobashi's. The main difference is that the sizes of tiles vary along the image: they are smaller near image details and larger otherwise (Figure 2d).

To obtain this result, the authors use centroidal Voronoi diagrams with a density function that depends on image features (edge magnitude in this case). Differently from Dobashi, they do not start from an hexagonal lattice, but the seeds of the first Voronoi diagram are found by sampling the image by using a quadtree [FB74]: the seeds are the centers of the leaf cells. A leaf is created when the color of its corresponding cell pixels are close to the average color of the cell. In particular, for each cell, they test if:

$$\max_{p \in C} d(I(p), c)^2 \leq \varepsilon, \tag{2}$$

where $I(p)$ is the color of the pixel p in C, c is the average color in C, and ε is a user-selected tolerance value. The image in Figure 2d has been obtained by using 2557 seed points, 10 iterations, $\varepsilon = 0.150$ and a minimum cell size equal to 36 pixels.

Figure 2 is provided to compare the above techniques in terms of visual appearance. Taking into account high-frequency details (edge features and their orientation) the general image structure is in some way preserved. Only Faustino and Figueiredo's approach makes use of tile size according to the different edge magnitude but without using the relative orientation.

A different approach is presented in [Mou03] where Mould proposes a technique to reproduce medieval stained glass windows. He presents an unsupervised method for transforming an arbitrary image into a stained-glass version (see Figure 3). The key issues in designing a stained glass window are tile boundaries and colors. He uses erosion and dilation operators to manipulate and smooth an initial region segmentation into a tiling. The algorithm chooses tile colors from the palette of heraldic tinctures and renders a displacement-mapped plane to obtain the final image. The algorithm can be summarized as follows:

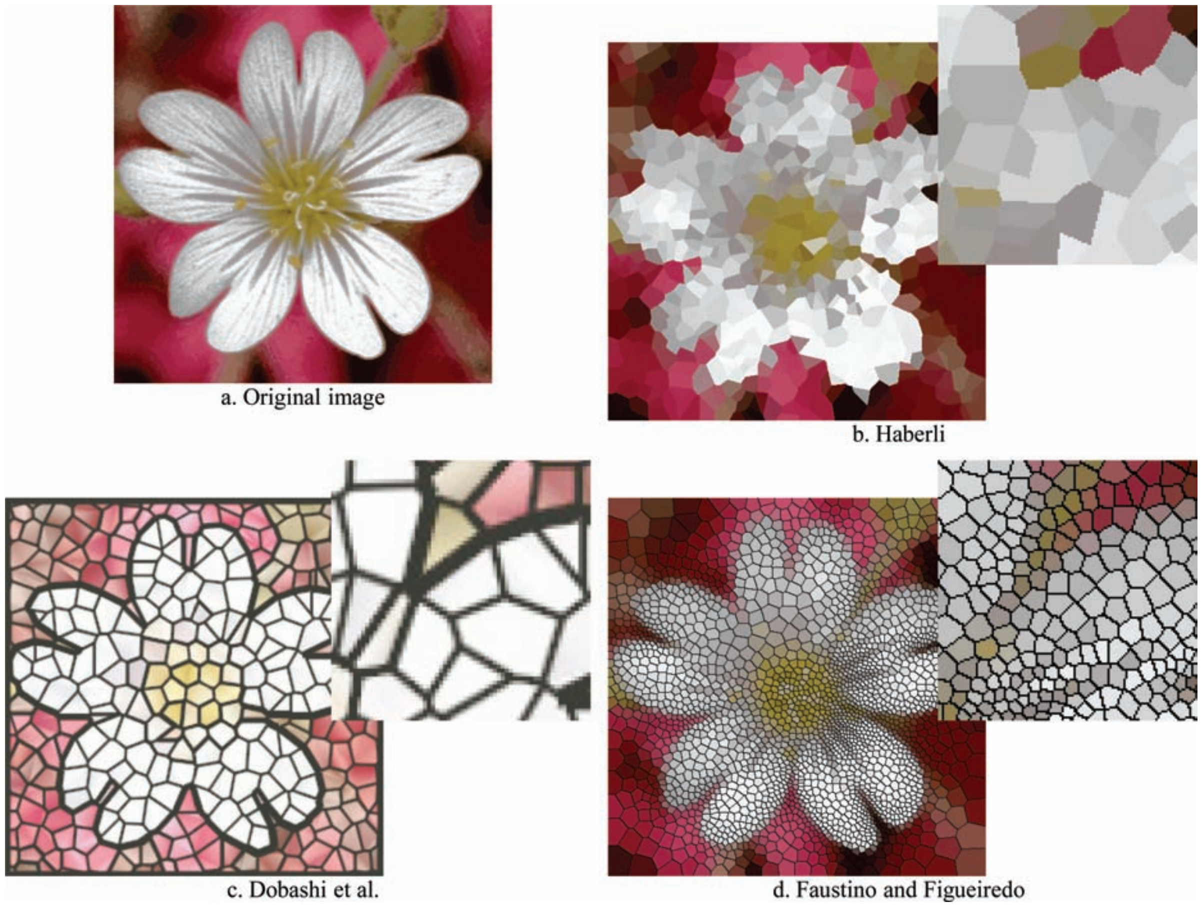


Figure 2: *Crystallization mosaics.*

1. obtain an initial segmentation of the image by using an image processing system (for example, he uses EDISON: [CEM02,CM02 and MG01];
2. evolve the segmentation to obtain an appropriate tiling having smooth boundaries and approximately convex pieces and lacking excessively large or excessively small pieces; the smoothing is obtained by the application of simple erosion and dilation operators from mathematical morphology;
3. choose a color for each tile; in particular it is possible to adopt the 'heraldic' palette: for a given tile, determine its average color in the original image and the distance of this color from each heraldic color; the tile is then colored with the nearest heraldic color;
4. apply a displacement map to a plane, representing the leading and irregularities in the glass;
5. render the result.

The method is able to reproduce stained-glass images in a very effective way. Only a few parameters are involved; the edge magnitude and relative orientation are implicitly considered by combining segmentation and morphological operators.

3. Ancient Mosaics

Since ancient times the art of mosaic has been extensively used to decorate public and private places. Today it is still possible to see some of such artistic works realized first by Greeks and Romans and later during the Byzantine Empire. Different kind of mosaics were produced also by old pre-Columbian people. Finally also in the modern area several artists have continued to deal with artistic mosaics [Mos06,Fio01].

Ancient mosaics are artworks constituted by cementing together small colored tiles. A smart and judicious use of orientation, shape and size may allow to convey much more information than the uniform or random 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 evidence may distract the observer from

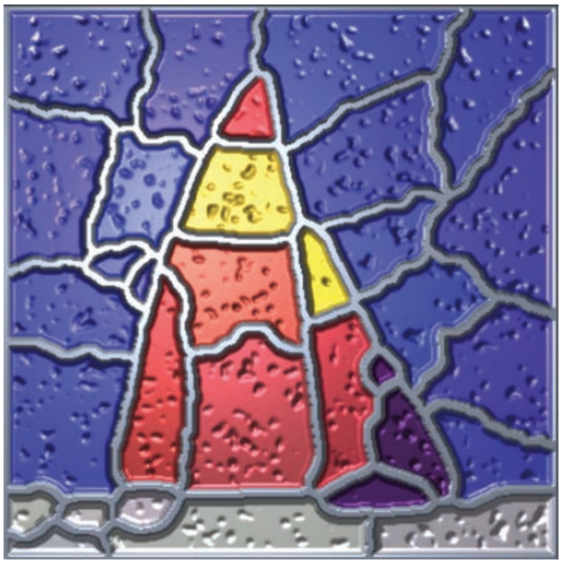


Figure 3: *Medieval stained glass by Mould.*

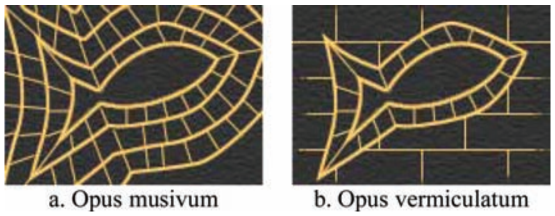


Figure 4: *Examples of ancient mosaics styles [Mos06].*

seeing the overall picture. To overcome such potential drawback, old masters placed tiles emphasizing the strong edges of the main subject to be represented. In our context we are not interested into physical design of a mosaic work (e.g. cementing materials, etc.), but in the way that the individual mosaic pieces - known to us as the *tesserae* - are laid down. By using different materials and/or combining the tesseraes in various ways, many different artistic styles and effects can be obtained. The general 'flow' of the mosaic is known as '*andamento*'. The typical ancient mosaics, today available in Computer Graphics, have a specific categorization in the field of Cultural Heritage. The old term 'opus' is used to describe the overall look of the mosaic. In particular the implemented techniques are the '*opus musivum*' and the '*opus vermiculatum*' (see Figure 4).

As the epithet of '*opus musivum*' says this means of 'quality worth of the muses', of great visual refinery and effect. '*Opus vermiculatum*' takes its name from the Latin for 'worm'. It refers to lines of tiles that snake around a feature in the mosaic. Often two or three rows of '*opus vermiculatum*' appear like a halo around something in a mosaic picture, helping it stand out from the background. The rendering of '*opus vermiculatum*' mosaics requires a clear separation between foreground and background because the two regions of the image have to be managed in different ways. The foreground region is covered as an '*opus musivum*', while the background region have to be covered by a regular grid of tiles (eventually perturbed by a random noise in size, position and rotation).

The first attempt to reproduce a realistic ancient mosaic was presented by Hausner [Hau01]. He proposed the mathematical formulation of the mosaic problem as described in Section 1. He obtained very good results using Centroidal Voronoi Diagrams (CVD), user selected edge features, *L1* (Manhattan) distance and graphic hardware acceleration (Figure 5b). In particular the method uses CVDs (which normally arrange points in regular hexagonal grids) adapted to place tiles in curving square grids. The adaption is performed by an iterative process which measures distances with the Manhattan metric whose main axis is adjusted locally to follow a chosen direction field (coming from the edge features). Computing the CVD is made possible by leveraging the z-buffer algorithm available in many graphics cards. Hausner's algorithm can be outlined as follows:

```
1. $S$ = list of random points on the input image;
2. while not converged:
     a. for each $p$ in $S$, place a square pyramid with apex at $p$;
     b. rotate each pyramid about the $z$ axis to align it to the field direction;
     c. render the pyramid with an orthogonal projection onto the $xy$ plane, producing a Voronoi diagram;
     d. compute the centroid of each Voronoi region;
     e. move each $p$ to the centroid of its region.
```

The number of iterations to reach convergence is one of the main drawbacks of this technique, mainly when there is no direct access to the graphic acceleration engine.

Another algorithm for the creation of ancient mosaics is presented in [DG05] and in [BDFG06]; this approach is based on directional guidelines and distance transform and leads to very realistic results (Figures 5c and 6b). The algorithm uses some known image processing techniques in order to obtain a precise tile placing that can be summarized as follows:

1. segment the image by using the Statistical Region Merging algorithm [NN04];
2. subdivide the image into background and foreground regions (optional);
3. for each pixel of the image evaluate the distance transform from the segmented region bounds;
4. evaluate the gradient matrix and the level line matrix;
5. place the tile.

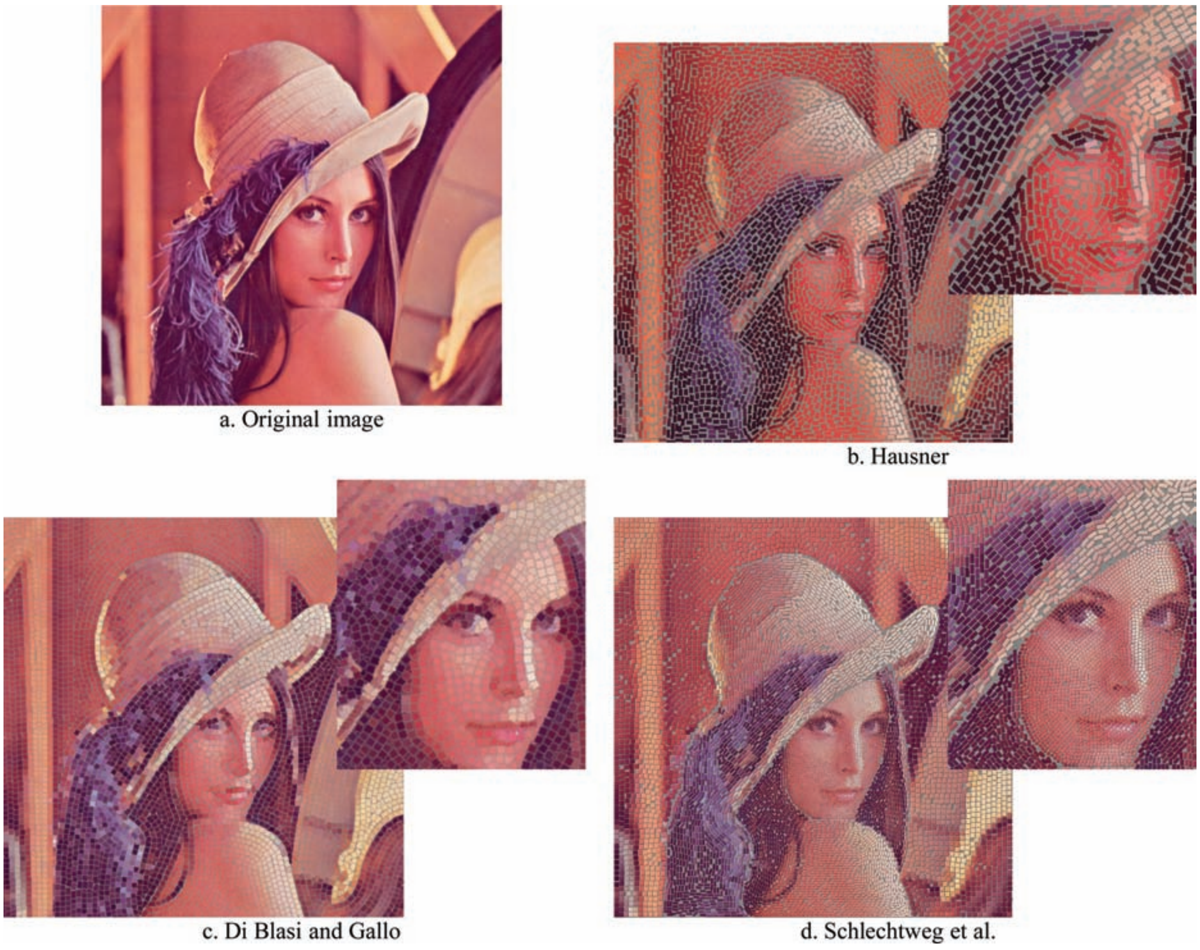


Figure 5: *Ancient Mosaics.*

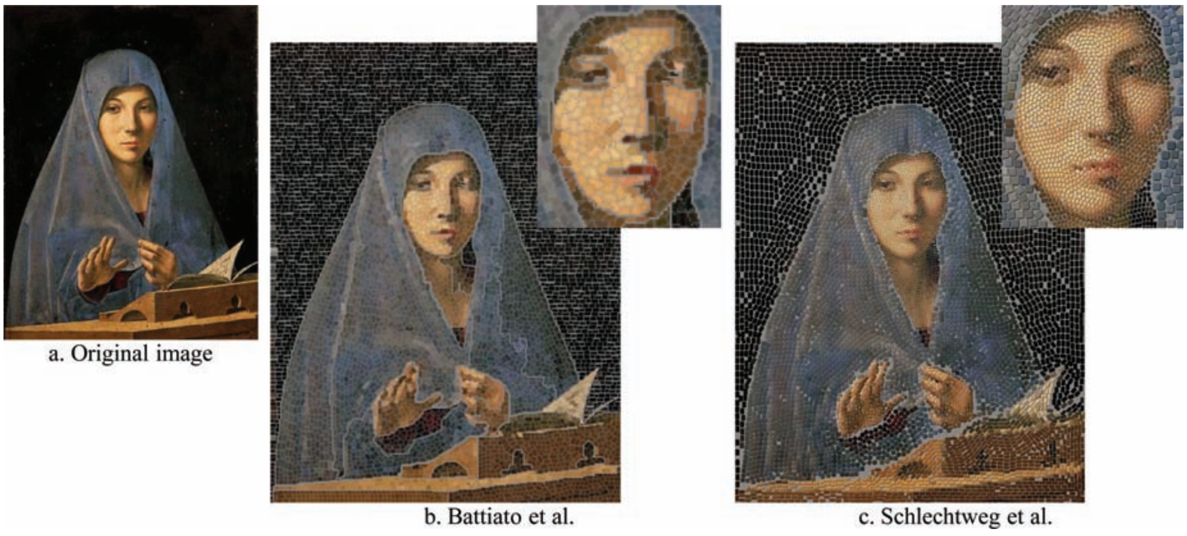


Figure 6: *Other examples of ancient mosaics.*

In particular point 5 can be described in more detail as follows:

5. while there are chains of pixels not yet processed in the level line matrix:
   a. select a chain;
   b. starting from an arbitrary pixel on it 'follow' the chain;
   c. place new tiles at regular distances along the path (the orientation of the tiles is assigned using the gradient information from matrix).

A high degree of similarity in terms of style with respect to ancient mosaics is clearly obtained. The overall asymptotic complexity is linear respect to the image size.

Recently, a novel technique for ancient mosaics generation has been presented in [SGS05]. The authors present an approach for stroke-based rendering that exploits multi-agent systems; they call the agents RenderBots. RenderBots are individual agents representing in general one stroke. They form a multi-agent system and undergo a simulation to disseminate themselves in the environment. The environment consists of a source image and possibly additional G-buffer support images (edge image, luminance image, etc.). The final image is created when the simulation is stopped by having each RenderBot executed its painting function. The complete mosaic generation process can be described as follows:

1. setup the environment (a number of RenderBots of a specific class are created and distributed in the environment);
2. distribute the RenderBots (randomly or interactively by the user);
3. while the image is not finished:
   a. simulate each bot (control of the bot physical behavior, computation of the new direction and velocity values and, possibly, change of the internal state of the RenderBot);
   b. move each bot (perform the actual movement of the bot by computing a new position);
   c. (eventually) paint each bot.

RenderBot classes differ in their physical behavior as well as their way of painting so that different styles can be created in a very flexible way. Thus they provide a unified approach for stroke based rendering. Different styles such as stippling, hatching, painterly rendering and mosaics can be created using the same framework. This is achieved by providing a specific class of RenderBots for each style.

Figures 5d and 6c show two images obtained by RenderBot; before generating the mosaics, the images had been manually segmented. This is necessary because MosaicBots orient themselves using the nearest edges. The generation of NPR-images using RenderBots is an (iterative) interactive process, so production-times depend on the artist's requirements and the desired quality. It took about an hour to produce the results presented here. The number of MosaicBots involved were approximately 9000 and 8000 respectively for the images in Figure 5d and Figure 6c.

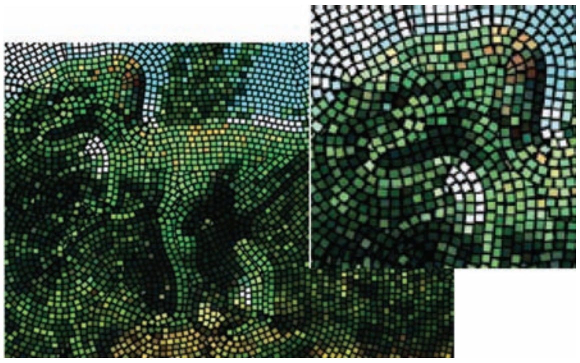


Figure 7: *Elber and Wolberg.*

A very advanced approach to the rendering of traditional mosaics is presented in [EW03] (Figure 7). 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 a very precise tile placement. Another point of this approach is the use of variable size tiles. Although the results are very good the technique seems limited to the case of a single, user-selected and closed edge curve.

A very interesting technique can be found in [FHHD05]; the authors present a new and efficient method to interactively create visually pleasing and impressive ancient mosaics. The algorithm is based again on the Lloyd's method for CVT (Centroidal Voronoi Tessellation) computation and can be viewed as a smart extension and/or optimization of the technique proposed by Hausner [Hau01]. They use a placement algorithm in an interactive fashion enabling the user to arrange tiles of various shapes and sizes. The user can easily control the distribution process by adding some other data such as contour lines and directional information. Tiles can be sized or shaped in order to better approximate the master image features. Additionally, this technique is less time expensive than using heuristic controlled automatic methods. An interactive tool is preferred because 'the proper arrangement of individual tiles is a highly artistic process' [FHHD05].

These authors claim that heuristic methods produce unwanted artifacts such as misaligned tiles. In more details this algorithm can be summarized as follows:

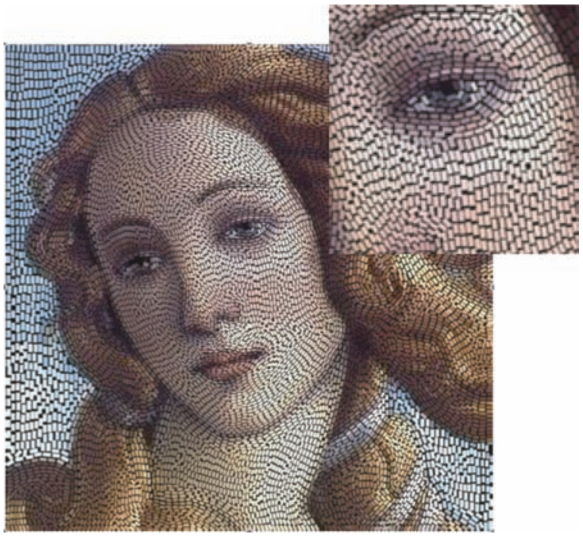


Figure 8: *Fritzsche et al.*

1. $M =$ set of randomly distributed and randomly oriented polygons P;
2. while movement/rotation is above a threshold value:
    a. for each P approximate its Voronoi region by using geometric modeled distance;
    b. perform the Principal Component Analysis on polygons and their Voronoi regions;
    c. compute the center of gravity CG of the polygons and of their Voronoi regions;
    d. move the polygons in order to match the CG of the polygons and the CG of their Voronoi regions;
    e. rotate polygons in order to align their principal component with the principal component of their Voronoi regions;

To create a mosaic a user must choose a master image and he has to define its feature lines and control polygons on the basis of the master image. He or she can hence choose the desired mosaic tile prototype (circles, quads or user-defined *n*-edges) and input the rough number of tiles to be inserted. After a preliminary unsupervised tile insertion he can manually insert/delete tiles by making use of interactive tools. Figure 8 shows a typical output image of this technique. This algorithm clearly outperforms all previously presented techniques in terms of aesthetic result. Unfortunately it requires a (crucial) user intervention and it is strictly dependent on the user's aesthetic skill and experience: two different users could obtain two totally different aesthetic results starting from the same input data (image, tile prototype, etc.).

To summarize this section the key of any technique aimed at the production of digital ancient mosaics is clearly the tile positioning and orientation. The methods presented in this section use different approaches to solve this problem, obtaining different visual results. The methods presented in [Hau01,EW03] and [FHHD05] are based on a CVD approach and on a global iterative tile positioning and orientation; the methods in [DG05] and [BDFG06] try to digitally reproduce the ancient artisans style by using a 'one-after-one' tile positioning and orientation and by introducing the idea of 'tile cutting' (typical of real ancient mosaic techniques), while in [SGS05] a non-deterministic approach is used.

4. Photo-mosaics

Photo-mosaic is one of the most interesting techniques (and one of the most re-discovered algorithms) in the field of digital mosaics. It transforms an input image into a rectangular grid of thumbnail images. In this approach the algorithm usually searches a large database of images for one that approximates a block of pixels in the main image. The resulting effect is very impressive. Even in this case no edge features are taken into account. Unfortunately many of the proposed algorithms to solve the photo-mosaic problem are not well documented.

Even before the computer's era, the process of making pictures from other pictures was well known. In 1973 Harmon [Har73] presented a work including several 'block portraits' (see Figure 9a). Harmon used these 'pixelated' portraits to study human perception and the automatic pattern recognition issues. For example he used them as a demonstration of the minimal condition to recognize a face. The image in Figure 9a is just a very low resolution rendering (252 pixels) of a gray image of Lincoln. Each pixel is seen as a 'tile'. This image of Lincoln needs to be viewed at a given distance to see the face. The fact that Lincoln's face is so well known makes its recognition easier.

In 1976 Dali [ND01] completed the painting in Figure 9b titled 'Gala contemplating the Mediterranean Sea, which at 30 meters becomes the portrait of Abraham Lincoln (Homage to Rothko)'. Note that Lincoln's face is made up by pictures with full tonal ranges, perhaps the earliest example of this technique. The nude taking up several tiles is Dali's wife Gala and one of the tiles is Harmon's gray scale image of Lincoln; so not only Dali did appropriate Harmon's Lincoln portrait into the overall composition, but he also reincorporated a smaller gray scale version into a single tile. If not the first, this is certainly one of the most sophisticated image made from other images.

In the 1970's the American artist Close [Clo70] began producing precisely gridded paintings (see Figure 9c). Some of these had a quality that reminds us of the Impressionists, while others seem to be computer generated or computer influenced.

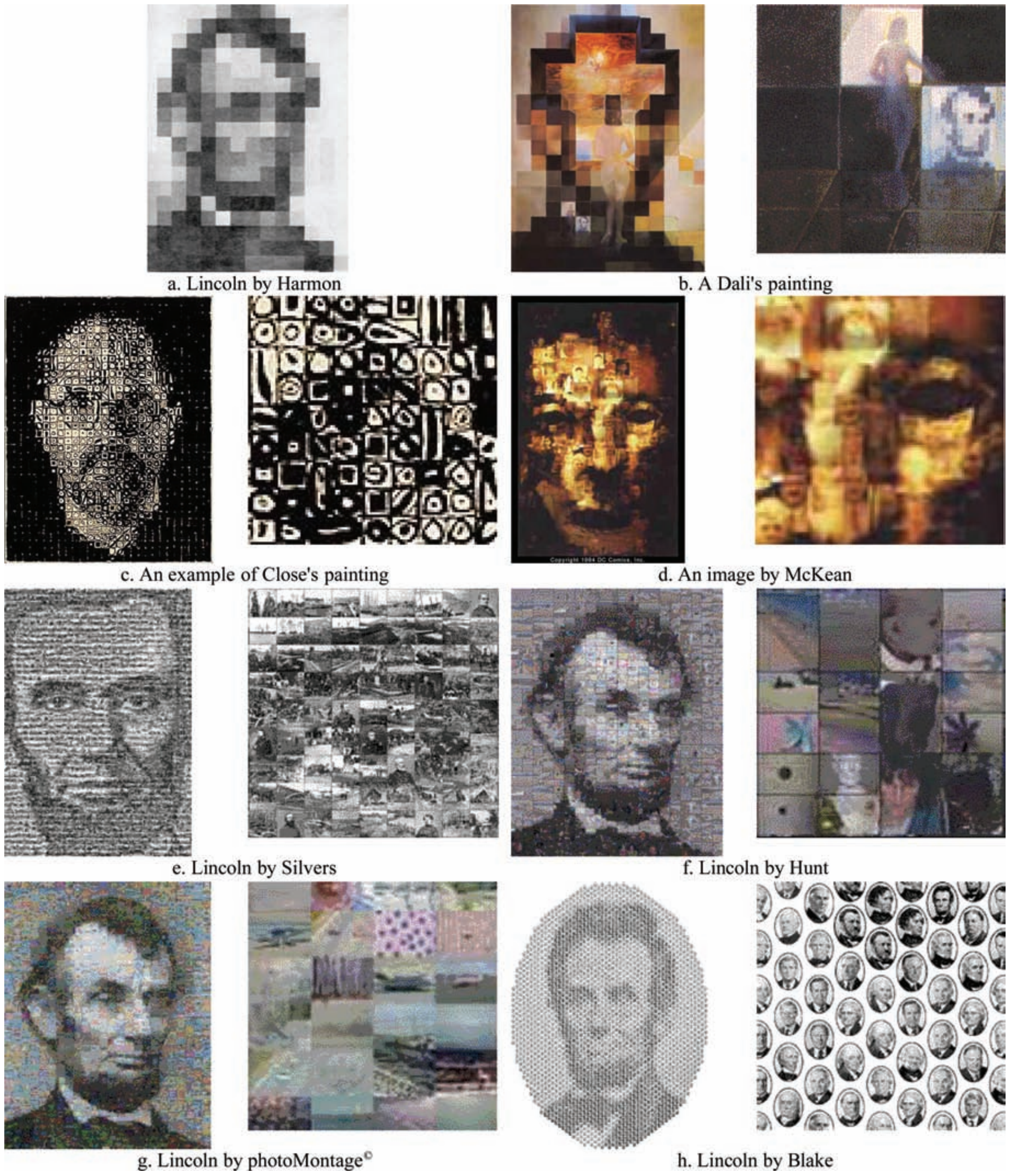


Figure 9: *Photo-mosaics techniques.*

The earliest example of making a tile larger than a single pixel came from an artist toying with Adobe Photoshop© [Ado06] and the earliest example of a photographic computer mosaic is the image created by McKean for Vertigo/DC comics in 1994 (see Figure 9d).

Computer replicated images made from tiled digital photographic pictures are recent because they require large computational resources. Silvers [SH97] began working on the first photo-mosaic while he was a graduate student at the MIT Media Lab. Each tile in his images represents much more than a single value. Smaller pictures match the overall image in tone, texture, shape and color. Silvers was commissioned by the US Library of Congress to create the portrait of Lincoln in Figure 9e using archived photos of the American Civil War. Today, Runaway Technology [Run06] (Silvers' company) produces photo-mosaic images as logos and illustrations for individuals, corporations and publications.

Hunt [Hun98], a computer programmer, created the image in Figure 9f by using three different size tiles to change the look of the grid.

ArcSoft photoMontage© [Arc06] is one of several commercial programs available nowadays (Figure 9g).

Scott Blake's image of Abraham Lincoln in Figure 9h is rendered by using 42 portraits of all US Presidents [Bla98]. He arranged the presidents according to their gray scale density. He offset the tiles on a beehive grid pattern to produce a hypnotic effect and used the oval portraits to fill the space in a more efficient manner.

Silver's original idea was successively extended by Klein *et al.* [KGFC02] to videos obtaining a video mosaic: a two-dimensional arrangement of small source video-tiles that suggests a larger video. To obtain the desired result they have to find a suitable space-temporal arrangement of videos that match a given large video maintaining a temporal coherence. In order to assess the match between source and target, they developed a distance measure based on average color and three-dimensional wavelet decomposition signatures in the YIQ color space. The most difficult task in creating video-mosaics is to resolve the tension between achieving good image matches and maintaining frame-to-frame coherence. To solve this task, they proposed a dynamic programming algorithm to automatically choose the smaller tiling subsequences from a large collection of candidate source video sequences. After the selection process, the color in the tiling videos is automatically adjusted. The general algorithm can be summarized as follows:

```
1. select a target video $T$ and a set of video-tiles $S$;
2. select a tiling grid (i.e. simple grid of rectangles or a
   more complex tiling pattern);
3. for each tile of the target video:
   a. for each frame of the target video:
      i.   select the best frame $S_j$ from the source videos
           matching the current tile in the current frame
           of $T$;
      ii.  color correct $S_j$ to mimic the current tile in the
           current frame of $T$;
      iii. paste $S_j$ into the tile for output.
```

Obviously, the key of the algorithm is the selection of the best frame from the source videos for each tile of the output video. To perform this task the authors take in account several features such as average color, distribution color, temporal and motion aspects and temporal coherence in the tiles. Figure 10 shows the typical output of this technique; it is also possible to download some videos directly from the website of the project [Vid06].

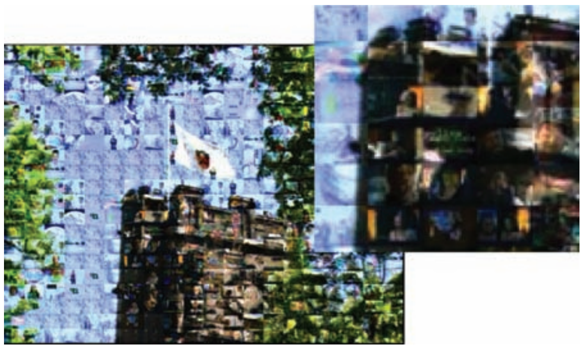


Figure 10: *An example of video mosaics.*

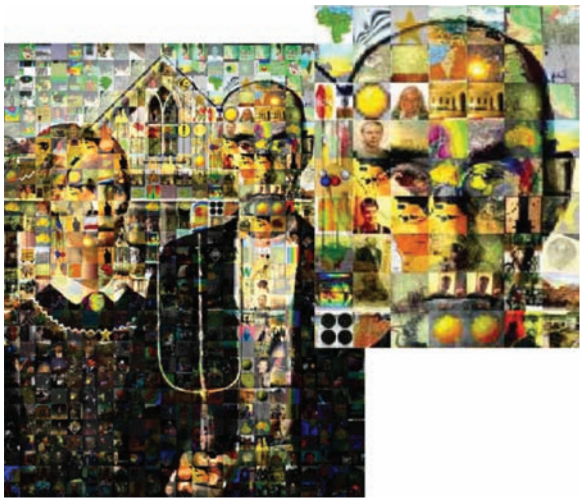


Figure 11: *An example of image mosaics.*

Mosaic animation is one of the more recent and promising direction for mosaicing even if the authors provide also several contributes in other related areas, in particular:

- a medium for layered video imagery;
- some simple and flexible metrics between video sequences;
- a framework to find optimal matching video subsequences while maintaining coherence.

In [FR98] the authors propose an extension of Silvers' ideas which first places the tile images and then alters their colors to better match the target image. The approach they

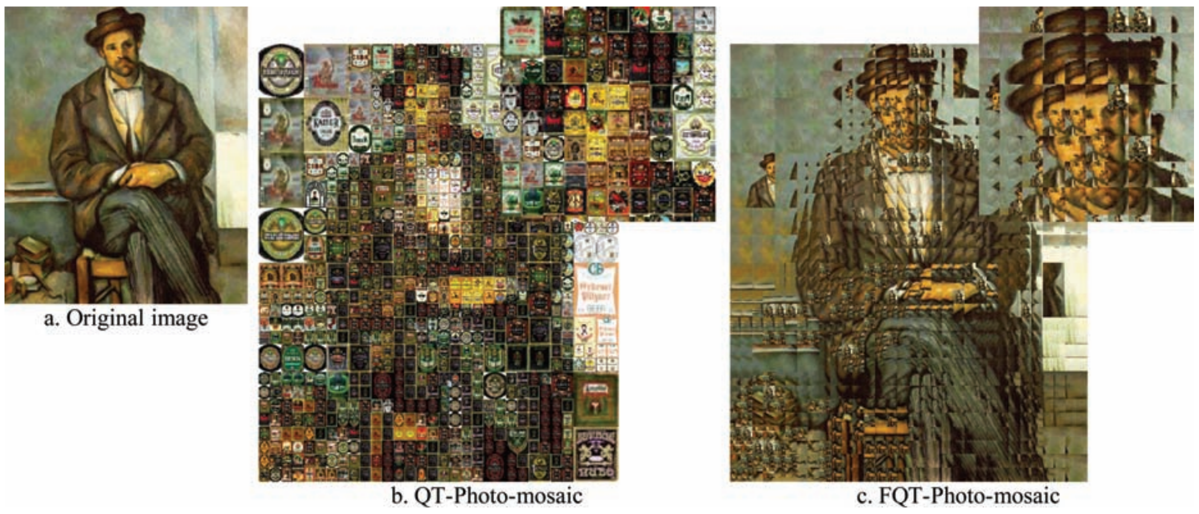


Figure 12: *An example of photo-mosaics by Di Blasi et al.*

took to alter the colors of the tile images is inspired by a dither matrix-based half-toning scheme.

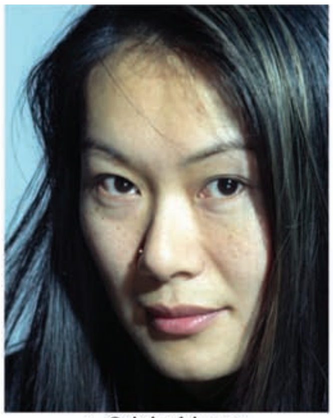
They call this technique 'image mosaics' (see Figure 11). Specifically, they make use of a correction rule, which takes as input an image tile and a desired average color a, and generates a correction function $F{:}R^1 \rightarrow R^1$ that maps a color x in the image tile to a color $F(x)$ in the final mosaic such that the region of the mosaic covered by the image tile will have the average color a. In particular, they use a combination of 'shifting' and 'scaling' rules: if they can use only a shift without sending any of the colors in the tile out of range, then they're done. If not, they shift as much as possible, then scale the resulting colors until the desired average is attained. Their 'shift-and-scale' rule can be summarized as follows:

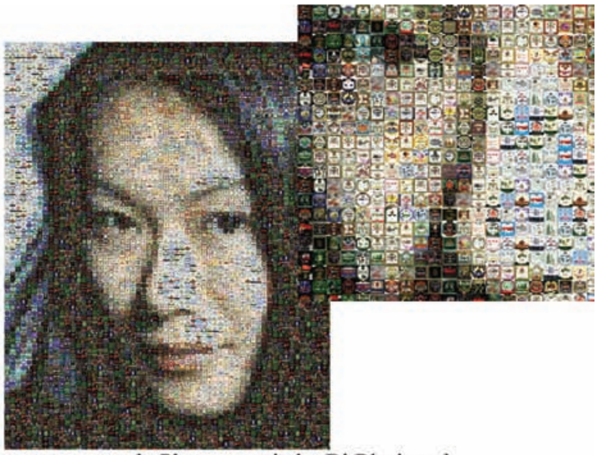
1. if the desired average color a is less than the average color a_t of the image tile, then:
   a. if the minimum color m_t of the image tile is greater than $a_t - a$, then use the shift rule $F(x) = x + (a - a_t)$;
   b. otherwise use a combination of shifting and scaling: $F(x) = a(x - m_t)/(a_t - m_t)$;
2. otherwise there is a symmetric pair of cases when the desired average a is greater than the tile average a_t.

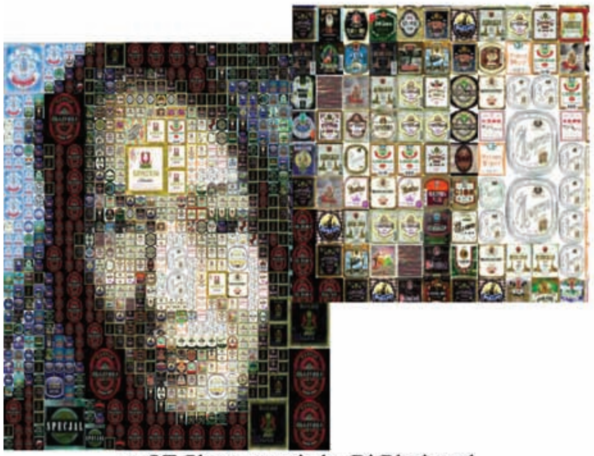
Di Blasi *et al.* [DP05,DGP06] presented an approach to speed up the search process based on the Antipole strategy [CFP*05]. The technique allows to obtain interesting effects in an efficient manner (see Figures 12, 13c and 13d). The Antipole Tree Data Structure is suitable for searches over large record sets embedded into a metric space (X, d). In particular, in this approach each picture in a database becomes a point in X; distance d between two pictures is computed in a straightforward way using the RGB values. Images are grouped into clusters of bounded radius by an efficient clustering algorithm: the Antipole Tree Clustering [CFP*05]. The clustering algorithm works in such a way that 'far' elements lie in different clusters. The algorithm is able to find, in linear time, a pair (A, B) (called Antipole), such that A and B are far apart. Then, elements of the set are partitioned according to their proximity to one of the two Antipole endpoints.

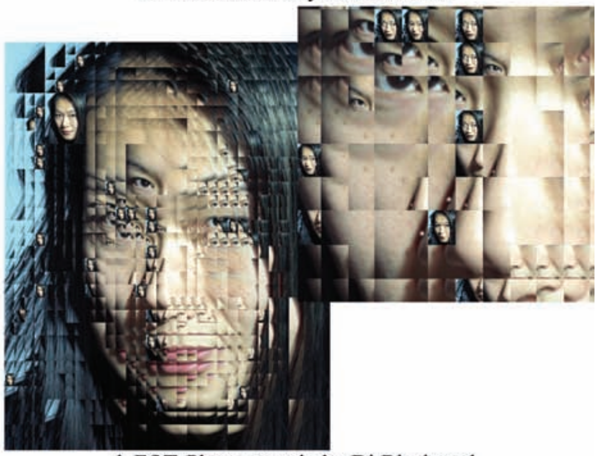
This splitting procedure is repeated recursively on can be thought as a starting point for the creation of a each subset and it will produce a binary tree whose leaves are the final clusters. The authors also propose some ideas to create other "photo-mosaic style" images. These ideas can be thought as a starting point for the creation of a general 'photo-mosaic framework'. They present two techniques called: QT-Photo-mosaic and FQT-Photo-mosaic. In QT-Photo-mosaic they use the quadtree splitting technique [FB74] to subdivide the input image into a non-regular rectangular grid. The splitting is based on the mean RGB values of the image and its variances. In FQT-Photo-mosaic they extend the quadtree idea in order to produce a fractal photo-mosaic image [BH93]. To obtain these results they restrict the database of thumbnail images to the image itself and its subQuadTree-images (see Figures 12, 13c and 13d).

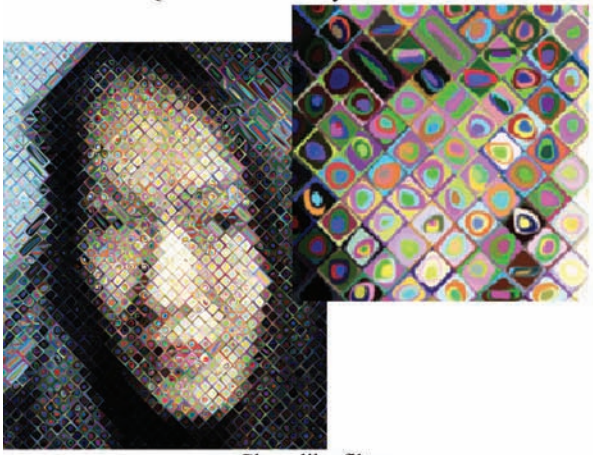
In [LSKL05], Luong *et al.* present a geometric technique for isoluminant color picking in linear color spaces proposing three methods to produce an improved pointlist filter, a 'Chuck Close'-like filter and a filter for stylized image mosaics. We will review only the last two methods (Figures 13e and 13f). Luong *et al.* observe that humans see world in two steps: first they process color and luminance information independently, next they merge such results to obtain the perceived final image. If two areas are isoluminant (that is they have the same luminance levels) they appear equal to our


a. Original image


b. Photo-mosaic by Di Blasi et al.


c. QT-Photo-mosaic by Di Blasi et al.


d. FQT-Photo-mosaic by Di Blasi et al.


e. Close-like filter

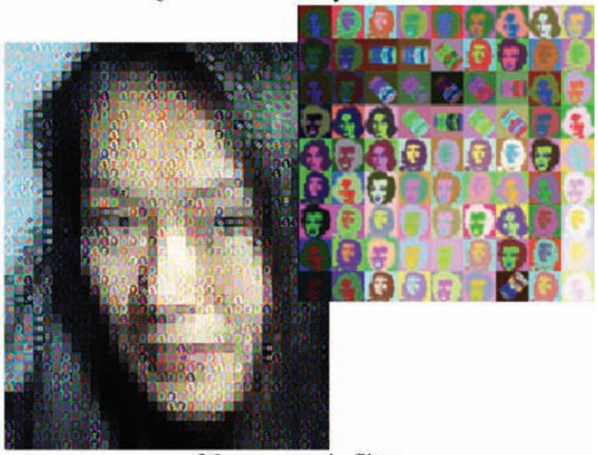

f. Image mosaic filter

Figure 13: *Examples of some photo-mosaic techniques.*

luminance perception system; this fact has been used by artists (for example Claude Monet) to obtain different effects. To reproduce the Chuck Close painting they first identify some typical features of Close's artworks:

a. each cell contains 1-2 nested quads and 3-10 nested blobs;
b. Close first fills the grid cell with an arbitrary color and then adds layers to move to the target color;

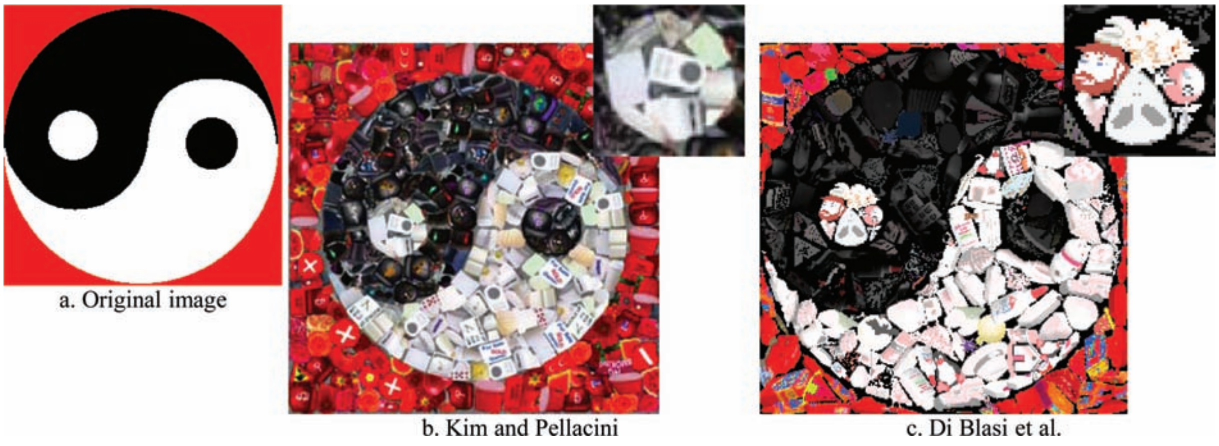


Figure 14: *Puzzle Image Mosaics.*

c. target cells will contain round blobs if there is no strong edge in the source cells, otherwise they will contain elongated (vertical, horizontal or diagonal) blobs;
d. diagonal blobs in adjoining cells with similar average colors may merge across the shared flat edges;
e. Close joins diagonal cells within large area of little or no edge information in order to eliminate uniformity in the visual field.

The authors approximate these properties by using the following algorithm:

1. filter the source image by a Canny edge detector;
2. compare each cell's edge magnitude with a fixed threshold in order to discriminate if the cell has a strong edge (features *c* and *d*);
3. randomly sample cells to check if they are in large uniform areas (features *e*);
4. using the edge information assign five layers (features *a*) of appropriate shapes (features *c*) to each cell, such that layers 1 and 3 are isoluminant, as are layers 2 and 4 (feature *b*);
5. create a grid of the source image;
6. for each cell in the grid evaluate its average color and dominant edge orientation;
7. if two cells with a shared edge have the same diagonal edge direction and similar average colors then join them;
8. randomly join cells in large, edge-less areas;
9. for each remaining (joined or not) cell:
    a. assign a set of nested blobs textures of the appropriate shape;
    b. evaluate the relative weight of each layer texture by using the isoluminant color picking algorithm;
    c. evaluate the layer colors;
    d. render the cell with each layer texture modulated by its corresponding layer color.

A typical image produced by this technique is presented in Figure 13e. They finally propose a method able to combine ideas from Silvers' algorithm [SH97] and their Close-like filter: Figure 13f shows an example of this novel idea. It is very interesting to compare the two above figures with Figure 9c which shows a Close's own artwork to verify the resemblance of the analogic and digital approaches.

5. Puzzle Image Mosaics

Puzzle Image Mosaic is inspired by Giuseppe Arcimboldo [Str99], a Renaissance Italian painter inventor of a form of painting called 'the composite head' where faces are painted not in flesh, but with rendered clumps of vegetables and other materials slightly deformed to better match the human features.

Kim and Pellacini [KP02] introduce a mosaicing technique, called Jigsaw Image Mosaic (JIM), where image tiles of arbitrary shapes are used to compose the final picture. The idea is quite similar to the photo-mosaic, but the final effect is very different and interesting (Figure 14b). The authors approach the problem by defining a mosaic as the tile configuration that minimizes a 'mosaicing energy function' and introduce a general energy-based framework for mosaicing problems that extends the algorithms presented in [Hau01] and in [SH97]. The energy function E used by Kim and Pellacini is defined as:

$$E = w_C \cdot E_C + w_G E_G + w_O E_O + w_D \cdot E_D \qquad (3)$$

where the color energy term E_C penalizes configurations that do not maintain the color of the input image. The gap energy term E_G penalizes configuration that have too much empty space in the final image, while a big overlap between tiles gives large overlap energy E_O. Finally, the deformation energy E_D penalizes configurations where tiles are highly deformed. The global E is then obtained by a linear combination obtained by using the weights w_i, $i = \{C, G, O, D\}$.

Another approach for the creation of the same kind of mosaics is presented in [DGP05]; this approach is based again on the Antipole strategy and leads to good results in an acceptable computation time (Figure 14c). The technique reformulates the problem as a search problem in a large database of small images taking into account some important features of the image to speed up the search process. Today, Magic Mosaics [Mag06] produces puzzle image mosaic posters and banners using a modified version of this software. To reach this goal the authors had to map a tile (shape and color) into the metric space X in order to create the Antipole data structure. The shape of a tile is composed by the pixels of the image having a non-transparent color. So, they perform the following steps:

1. evaluate the shape's mass center;
2. subdivide the shape into 90 segments, obtaining 90 vertices;
3. compute the (Euclidean) distance of each vertex from the mass center and normalize the value in [0,1] (the normalization is performed in order to make the distances 'scale independent').

This leads to a vector x made up of 90 components: x is the feature vector of the image in the data structure. The shape distance is computed evaluating the Euclidean distance between feature vectors. The computation takes into account all the possible shifting between the two arrays (that is all the possible mutual rotations of the two shapes). This operation is performed in order to make the distance 'rotation independent' and 'starting point independent'; since a shape is subdivided into 90 segments a rotation error of at most 4 degrees ($\pi/45$ radians) is committed. The final algorithm can be summarized as follows:

1. start with an input image;
2. perform the directional guideline detection by using the same technique proposed in [DG05];
3. perform the morphological operation 'dilate' obtaining the image G (the dilation is performed only for better aesthetic results and it does not affect the subsequent steps);
4. compute a Voronoi diagram V of the same size of the input image (the set of points is randomly chosen and its cardinality is user selected);
5. merge the images G and V obtaining the image R;
6. for each region R_i of R:
    a. perform the algorithm described above in order to obtain the feature vector x of R_i (x can hence be used to perform the search in the Antipole tree);
    b. perform the best matching;
    c. perform a color shifting in order to align the median color of the selected tile with the median color of R_i;
    d. rotate and resize the tile to fit and paint it over the region.

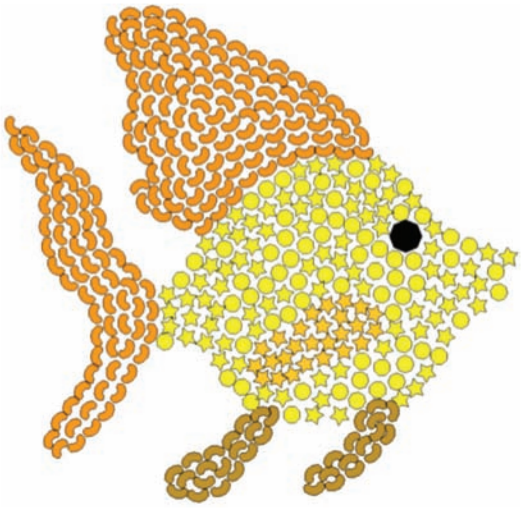


Figure 15: *A frame from an example of animosaic.*

An animated version of Puzzle Image Mosaics is presented in [SLK05] (Figure 15). Animated mosaics (or 'Animosaics') are techniques of stop-motion animation created by arranging small irregular objects frame by frame. The manual process of placing and moving the objects in each frame is time-consuming and laborious; for this reason, the authors propose an approach that extends the ideas presented in [Hau01] and [KP02] to temporal coherence to create mosaic animations. In particular, they characterize the temporal coherence as the coordinated movement of groups of primitives and describe a method to achieve this movement. They use a Centroidal Area-based Voronoi Diagram (CAVD) which extends Hausner's work to support multiple, arbitrarily shaped tiles. The CAVD approach is almost identical to the PCA-based technique presented in [FHHD05]. The proposed algorithm can be summarized as follows:

1. take as input an animated scene represented as a collection of 2D containers and a set of irregular tiles;
2. for each container (manually):
    a. pick the desired tile shapes and sizes;
    b. pack the container's first frame;
3. generate the remaining frames of the container's packing by:

   a. automatically advect the container's tiles from the current frame to the next to promote temporal coherence;
   b. manually insert in the new frame new tiles and refine the current packing to reflect container changes.

Figure 15 shows a typical frame of an output of an Animosaic.

The same authors extend in [DKLS06] their technique by merging the speed and flexibility of Voronoi based approaches [Hau01] with the robustness and convergence properties of an energy optimization function [KP02]. In particular, they propose a mosaic packing technique by utilizing an energy function as part of a fast, efficient tile adjustment algorithm. The energy combines an overlap penalty with the sum of squared distances (*SSD*) between the points in a tile's Voronoi region and the tile's perimeter. More precisely, let p be a pixel in an image I and let t be a specific tile from a set of tiles T. They define a metric:

$$SSD(t) = \sum_{p \in I} dist(p, t)^2 \cdot f(p, I(t)), \qquad (4)$$

where $dist(p, t)$ is the shortest (Euclidean) distance from p to t's perimeter on a normalized $[0, 1]^2$ canvas, $I(t)$ denotes pixels in t's Voronoi region and $f(p, I(t))$ evaluates to 1 if $p \in I(t)$ and 0 otherwise. The proposed algorithm can be summarized as follows:

```
1. while not converged:
   a. compute the area Voronoi diagram using all tiles;
   b. for each tile $t \in T$, move $t$ to minimize $SSD(t)$.
```

To minimize *SSD* it is necessary to evaluate it over all pixel shift of t. This can be done quickly in the following way:

1. create a distance image D in which each pixel contains the squared distance to tile t;
2. create a Boolean image V, which has values of either 0 or 1 at each pixel indicating whether that pixel is in t's Voronoi region
3. find the shift of t that minimizes the dot product of Equation 4 by performing an image correlation between D and V.

This last step can be solved efficiently by using the Fast Fourier Transform (FFT):

$$R = FFT^{-1}(FFT(D) \cdot \boldsymbol{FFT}(V)), \qquad (5)$$

where the 'tilde' symbol indicates the complex conjugate; then it is possible to scan R in order to find the pixel location with minimum value, that is the best shift of t. Using the same technique the authors also minimize the overlap function defined as follows:

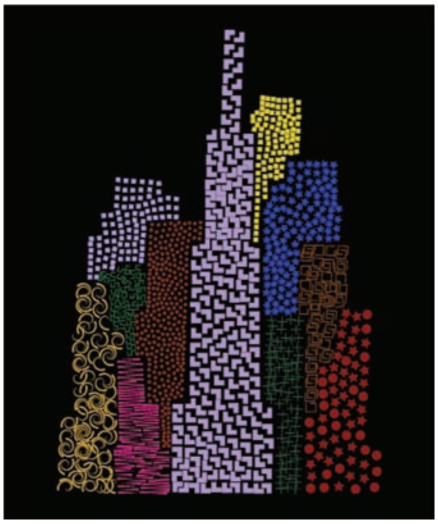


Figure 16: *A frame from an example of the technique proposed by Dalal et al.*

$$Overlap(t) = w \cdot \sum_{p \in I} O_t(p) \cdot f(p, t), \qquad (6)$$

where $O_t(p)$ is the number of tiles other than t covering pixel p, w is a weighting factor and $f(p, t)$ evaluates to 1 if $p \in t$ and 0 otherwise. Finally the authors define the energy function to minimize:

$$E = \sum_{t \in T} SSD(t) + \frac{1}{2} \cdot Overlap(t), \qquad (7)$$

This approach does not consider tiles orientation. To solve this problem the process is repeated for each (discrete) orientation selecting the minimum value over all position+orientation combinations. Figure 16 shows a frame from an example of this technique. Finally the authors prove that each iteration of their optimization algorithm can be performed in $O(r \cdot n \cdot lg(n(m)))$, where n is the number of pixels, m is the number of tiles and r is the number of possible rotations.

6. Final Discussion

In this Section we briefly summarize the various characteristics of each presented method. Table 1 shows in a compact way, grouped by category, some specific details. The overall computational complexity is reported with respect to the number n of pixels of the input image, the number m of involved thumbnail images (for multi picture mosaics) or videos (for animated mosaics), the number k of iterations for non deterministic methods, the number r of rotations (for [DKLS06]) and the number f of frames (for [KGFC02]). For

Table 1: *List of the presented digital mosaic approaches and related features.*

	Method	**Computational Complexity** (n = number of pixels) (m = # of thumbnails) (k = # of iterations) (r = # of rotations) (f = # of frames)	Size	Partitioning	Deterministic	Iterative	Interactive
Crystallization mosaics	Haeberli [Hae90]	$O(n \cdot lg(n))$	Fixed	Voronoi	No	No	No
	Dobashi et al. [DHJN02]	$O(k \cdot n \cdot lg(n))$	Fixed	Voronoi	No	Yes	No
	Mould [Mou03]	$O(n)$	Variable	Segmentation	Yes	No	No
	Faustino and Figueiredo [FDF05]	$O(k \cdot n \cdot lg(n))$	Variable	Voronoi	No	Yes	No
Ancient mosaics	Hausner [Hau01]	$O(k \cdot zBufferingExecutionTime)$	Variable	CVD	No	Yes	No
	Elber and Wolberg [EW03]	$O(k \cdot n \cdot lg(n))$	Variable	CVD	No	Yes	No
	Battiato et al. [BDFG06]	$O(n)$	Fixed	None	Yes	No	No
	RenderBot [SGS05]	$O(k \cdot botExecutionTime)$	Fixed	None	No	Yes	Yes
	Fritzsche et al. [FHHD05]	$O(k \cdot n \cdot lg(n))$	Variable	CVD	Yes	Yes	Yes
Photo-mosaics	Image Mosaic [FR98]	$O(m \cdot n)$	Variable	Grid	Yes	No	No
	Close-like Mosaic [LSKL05]	$O(m \cdot n)$	Fixed	Grid	Yes	No	No
	Di Blasi et al. [DGP06]	$O(n \cdot lg(m)$	Variable	Grid	Yes	No	No
	Video mosaics [KGFC02]	$O(n \cdot f^2 \cdot m)$	Fixed	Grid	Yes	No	No
Puzzle image mosaics	Kim and Pellacini [KP02]	$O(k \cdot n^2)$	Fixed	Voronoi	No	Yes	No
	Di Blasi et al. [DGP05]	$O(n \cdot lg(m))$	Fixed	Voronoi	Yes	No	No
	Animosaics [SLK05]	$O(k \cdot n \cdot lg(m))$	Variable	CVD	Yes	Yes	Yes
	Dalal et al. [DKLS06]	$O(k \cdot r \cdot n \cdot lg(n/m))$	Variable	CVD	Yes	Yes	No

iterative methods the parameter k is not always known *a priori*: usually a suitable tuning phase is required. Note that for sake of clarity in some techniques we consider the grid size and/or granularity g proportional to the number of pixels n of the input image. To have a low computational complexity (almost linear in the number of involved pixel) is fundamental to be able to reproduce high-resolution mosaics in both categories (tile, multi-picture) without requiring expensive hardware resources.

The remaining columns in Table 1 list other parameters and methodologies. One of the main effects of using fixed or variable tile size is the different emphasis given to specific details of the input image; the variability of the tile sizes produces a degradation in size along edges and textured areas especially if compared with classical mosaic appearance. Of course the particular partitioning strategy used by each method impacts the final result even if the tile positioning (orientation, cutting, etc.) is more crucial especially for unsupervised techniques (i.e. the key factor of each proposed strategy).

No evaluation is given in terms of 'aesthetic pleasure': no objective metrics are available to measure the effectiveness (accuracy of any digital mosaic. Aesthetic evaluation of any work produced by using supervised or unsupervised CA techniques is not so easy, because any objective metric is clearly inadequate [Col04]. Only an artist could give a more reliable, but subjective judgment. The aesthetics of an output could be further evidenced by non-scientific and non-academic interests; for example the photo-mosaic algorithm created by Silver [SH97] is today used by Runaway Technology [Run06] (Silvers' company) to produce photo-mosaic images as logos and illustrations for individuals, corporations and publications, Magic Mosaics [Mag06] produces puzzle image mosaic posters and banners using a modified version of the software proposed by Di Blasi *et al.* [DGP05] and the software proposed by Di Blasi and Gallo [DG05] has received interest from many artists, companies and Fine Arts Academies.

There are several ways to improve the aesthetic of the results and several ideas started from these works. In particular:

for ancient mosaics

- a different strategy for choosing, in case of overlapping tiles, which tile has to be cut; heuristic rules or, perhaps, randomized choices could produce different outcomes;
- generalization of the 'mosaicists' heuristic' to other kinds of primitive-based non-photorealistic image processing seems possible and quite promising;
- some generalizations proposed in [EW03], such as variable size tiles and photo-mosaic, are also considered for future work and research; it is also interesting to explore the possibilities offered using different basic shapes than rectangular tiles;

for photo-mosaic and puzzle image mosaic

- the use of Antipole Tree or other data structures in other fields of NPR to speed-up the rendering process;

for all presented methods

- automatic optimized choices of tile scale relative to each input image is an open problem worth further investigation;
- the extension of mosaic techniques to other kinds of mosaics as proposed in [EW03];
- a different method to better find the directional guidelines is an important research investigation issue;
- exploitation of hardware graphics primitives to accelerate the mosaic synthesis;
- extension of the proposed methods for mosaic rendering of 3D surfaces is probably the most exciting direction of research.

7. Final Summary

In this paper, we surveyed several approaches to NPR of digital images in the field of mosaic generation. The various methods have been grouped together according to the main features. In particular, we singled out four different kind of mosaics: '*crystallization*' mosaics, '*ancient*' mosaics, '*photo*'-mosaics and '*puzzle image*' mosaics. It is also possible to group the mosaic creation methods by using different criteria. The common and different ideas among the methods had been reported and described. The various techniques had been compared also with respect to the overall performances both in terms of achieved visual effects and computational complexity.

Acknowledgments

We thank the anonymous reviewers for their insightful comments and in help us improve the paper. We also sincerely thank Tobias Germer, Stefan Schlechtweg, Thomas Strothotte, Fabio Pellacini, Junhuan Kim, Geisa Martins Faustino and Luiz Henrique de Figueiredo for the precious hints and the technical support that they have provided us.

References

[Ado06] Adobe Photoshop, http://www.adobe.com, 2006.

[Arc06] ArcSoft PhotoMontage, http://www.arcsoft.com, 2006.

[BDFG06] BATTIATO S., DI BLASI G., FARINELLA G. M., GALLO G.: A Novel Technique for Opus Vermiculatum Mosaic Rendering. In *Proc. ACM/WSCG2006* (2006).

[BH93] BARNSLEY M. F., HURD L. P.: *Fractal Image Compression*. AK Peters Ltd., 1993.

[Bla98] BLAKE S., http://www.barcodeart.com/, 1998.

[CAS*97] CURTIS C. J., ANDERSON S. E., SEIMS J. E., FLEISCHER K. W., SALESIN D. H.: Computergenerated watercolor. In *Proc. SIGGRAPH1997* (1997), 421–430.

[CFP*05] CANTONE D., FERRO A., PULVIRENTI A., REFORGIATO RECUPERO D., SHASHA D.: Antipole Tree indexing to support range search and K-nearest neighbor search in metric spaces. *IEEE/TKDE 17*, 4 (2005), 535–550.

[CGM02] CHRISTOUDIAS C., GEORGESCU B., MEER P.: Synergism in low-level vision. In *Proc. ICPR2002* (2002), 150–155.

[CH03] Collomosse J. P., Hall P. M.: Cubist Style Rendering from Photographs. *IEEE/TVCG 9*, 4 (2003), 443–453.

[Clo70] Close C., http://www.chuckclose.coe.uh.edu/, 1970.

[CM02] Comanicu D., Meer P.: Mean shift: a robust approach toward feature space analysis. *IEEE/TPAMI 24*, 4 (2002), 603–619.

[Col04] Collomosse J. P.: Higher Level Techniques for the Artistic Rendering of Images and Video. *Ph.D Thesis* (2004).

[Com06] Computational Aesthetics in Graphics, Visualization and Imaging. http://www.computationalaesthetics.org/, 2006

[DBVKO97] De Berg M., Van Kreveld M., Overmars M., Schwarzkopf O.: *Computational Geometry*. Springer-Verlag, Berlin, 1997.

[DG05] Di Blasi G., Gallo G.: Artificial Mosaics. *The Visual Computer 21*, 6 (2005), 373–383.

[DGP05] Di Blasi G., Gallo G., Petralia M.: Puzzle Image Mosaic. In *Proc. IASTED/VIIP2005* (2005).

[DGP06] Di Blasi G., Gallo G., Petralia M.: Smart Ideas for Photomosaic Rendering. In *Proc. Eurographics Italian Chapter 2006* (2006), 267–271.

[DHJN02] Dobashi J., Haga T., Johan H., Nishita T.: A Method for Creating Mosaic Images Using Voronoi Diagrams. In *Proc Eurographics2002* (2002), 341–348.

[DHVO00] Deussen O., Hiller S., Van Overveld K., Strothotte T.: Floating points: A method for computing stipple drawings. In *Proc. Eurographics2000* (2000), 40–51.

[DKLS06] Dalal K., Klein A.W., Liu Y., Smith K.: A Spectral Approach to NPR Packing. In *Proc. NPAR2006* (2006), 71–78.

[DP05] Di Blasi G., Petralia M.: Fast Photomosaic. In *Poster Proc of ACM/WSCG2005* (2005).

[EW03] Elber E., Wolberg G.: Rendering Traditional Mosaics. *The Visual Computer 19*, 1 (2003), 67–78.

[FB74] Finkel R.A., Bentley J.L.: Quad trees: A data structure for retrieval on composite keys. *Acta Informatica 4* (1974), 1–9.

[FDF05] Faustino G. M., De Figueiredo L. H.: Simple Adaptive Mosaic Effects. In *Proc. SIBGRAPI2005* (2005), 315–322.

[FHHD05] Fritzsche L. P., Hellwig H., Hiller S., Deussen O.: Interactive design of authentic looking mosaics using Voronoi structures. In *Proc. 2nd International Symposium on Voronoi Diagrams in Science and Engineering VD 2005 Conference* (2005)

[Fio01] Fiorentini Roncuzzi I.: *Mosaic. Materials, Techniques and History*. MWeV Editions, 2001.

[For87] Fortune S.: A Sweep-line algorithm for Voronoi diagrams. *Algorithmica* 2, (1987), 153–174.

[FR98] Finkelstein A., Range M.: Image mosaics. In *Proc. RIDT1998* (1998), 11–22.

[Hae90] Haeberli P.: Paint by Numbers: Abstract Image Representation. In *Proc. SIGGRAPH1990* (1990), 207–214.

[Har73] Harmon L.D.: The Recognition of Faces. *Scientific American 229*, 5 (1973), 70–82.

[Hau01] Hausner A.: Simulating Decorative Mosaics. In *Proc. SIGGRAPH2001* (2001), 573–580.

[HHD03] Hiller S., Hellwig H., DeKussen O.: Beyond Stippling-Methods for Distributing Objects on the Plane. In *Proc. Eurographics2003* (2003), 515–522.

[HJO*01] Hertzmann A., Jacobs C., Oliver N., Curless B., Salesin D.H.: Image Analogies. In *Proc. SIGGRAPH2001* (2001), 327–340.

[Hun98] Hunt W. L. http://home.earthlink.net/w̃lhunt/, 1998.

[KGFC02] Klein A. W., Grant T., Finkelstein A., Cohen M. F.: Video Mosaics. In *Proc. NPAR2002* (2002), 21–28.

[KP02] Kim J., Pellacini F.: Jigsaw Image Mosaics. In *Proc SIGGRAPH2002* (2002), 657–664.

[LSKL05] Luong T. Q., Seth A., Klein A. W., Lawrence J.: Isoluminant Color Picking for Non-Photorealistic Rendering. In *Proc. of Graphics Interface 2005* (2005), 233–240.

[Mag06] Magic Mosaics, http://www.magicmosaics.com/, 2006.

[MG01] Meer P., Georgescu B.: Edge detection with embedded confidence. *IEEE/TPAMI 23*, 12 (2001), 1351–1365.

[Mos06] Mosaic Art in Vitreous Glass, Millefiori, Tesserae Mosaics by Shelby Glass Studio, http://www.mosaic-tile-art.com/mosaic.html, 2006.

[Mou03] MOULD D.: A Stained Glass Image Filter. In *Proc. 14th Eurographics Workshop on Rendering 2003* (2003), 20–25.

[ND01] NERET G., DESCHARNES R.: *Dali: The Paintings*. Taschen, 2001.

[NN04] NOCK R., NIELSEN F.: Statistical Region Merging. *IEEE/TPAMI 26*, 11 (2004), 1452–1458.

[PS87] PREPARATA F. P., SHAMOS M. I.: *Computational Geometry: An Introduction*. Springer-Verlag, New York, 1987.

[Run06] Runaway Technology, http://www.photomosaic.com/, 2006.

[Sec02] SECORD A.: Weighted Voronoi stippling. In *Proc. NPAR2002* (2002), 37–43.

[SGS05] SCHLECHTWEG S., GERMER T., STROTHOTTE T.: RenderBots – Multi-Agent Systems for Direct Image Generation. *Computer Graphics Forum 24*, 2 (2005), 137–148.

[SH97] SILVERS R., HAWLEY M.: *Photomosaics*. Henry Holt, 1997.

[SHS02] SECORD A., HEIDRICH W., STREIT L.: Fast primitive distribution for illustration. In *Proc. 13th Eurographics Workshop on Rendering* (2002), 215–226.

[SLK05] SMITH K., LIU Y., KLEIN A. W.: Animosaics. In *Proc. Eurographics symposium on Computer animation* (2005), 201–208.

[ST90] SAITO T., TAKAHASHI T.: Comprehensible rendering of 3-d shapes. In *Proc SIGGRAPH1990* (1990), 197–206.

[Str99] STRAND C.: *Hello, Fruit Face!: The Paintings of Giuseppe Arcimboldo*. Prestel, 1999.

[Vid06] Video Mosaics, http://www.cs.princeton.edu/~awklein/vidmosaics.html, 2006.