Image Mosaics with Irregular Tiling

Lina Zhang
State Key Lab. of CAD&CG
Zhejiang University
Hangzhou, China
zhanglina@zjucadcg.cn

Jinhui Yu
State Key Lab. of CAD&CG
Zhejiang University
Hangzhou, China
jhyu@cad.zju.edu.cn

Abstract

We present an approach for generating image mosaics with irregular tiles made up from patches taken from photographs, paintings and texture images. We propose a method to generate irregular tiling patterns using polygon tessellation in conjunction with a feature-based segmentation scheme, so that features in the input image can be better preserved in the generated mosaics. In order to avoid the mismatch in roughness between the sub-image in the tile region of the input image and tile images in the tile dataset that may arise in the previous RGB color based image descriptors, we introduce the concept of region entropy into the image descriptor to achieve better match in both color and roughness. A few mosaic images generated by our system are presented, some of which have effects of Chinese characteristics.

1. Introduction

Mosaics, as an ancient art form, is usually made by arranging small pieces of stone or glass to create a picture or pattern. Mosaics may use either regular tiles such as cubic stones, or irregular shapes and sizes of ceramic, porcelain and glass for greater design variety. In computer generated image mosaics, various elements such as photographic images and icons can be used as tiles. Compared with traditional mosaics, image mosaic tiles may convey some additional visual information to the viewers, thus they enrich the expressive power of mosaics and have many interesting applications.

In this paper we address the problem of creating image mosaics as follows: given a collection of tile images *T* and a target image *I*, construct an image mosaic *M* with irregular tiles containing images selected from *T* which will resemble *I*. Fig.1 shows three examples of image mosaics of a picture of panda generated by our system. The main contributions of our work are:

(1) Most image mosaics use equally spaced rectangular (including square) image tiles, a few use rectangular image tiles which change their sizes adaptively in the salient and non-salient regions as detected by the variations of RGB colors in *I*. Inspired by hand-made glass mosaics (Fig.2 *right*), we propose an irregular tiling scheme using a polygonal tessellation which is then modified by use of edge information obtained from image segmentation.

(2) Currently color information is used to match a tile region in *I* and tile images in *T* in most image mosaic systems. However, such color based image descriptors may lead to a mismatch in roughness between the two, so we introduce an image descriptor that takes both color and region entropy into account and use it to match the tile region in *I* and tile images in *T*.

2. Related work

During the past one and a half decades a large amount of work has been applied to digital mosaicing. A nice overview of mosaicing methods was recently presented by Battiato *et al.* [1]. Haeberli [2] first used Voronoi diagrams to tessellate the image with tiles of variable shapes and it did not attempt to follow edge features. Then Dobashi *et al.* [3] extended the original idea of Haeberli [2] and generated aesthetically more pleasant results because they integrated edge information with Voronoi tessellation. Faustino and Figueiredo [4] presented a technique similar to Dobashi's, but the sizes of tiles were smaller near image details and larger otherwise. Hausner [5] obtained very good results of ancient mosaics using Centroidal Voronoi Diagrams. A very advanced approach to the rendering of traditional mosaics was presented by Elber *et al.* [6]. This technique was based on offset curves that got trimmed-off the self intersecting segments with the guidance of Voronoi diagrams. Fritzsche *et al.* [7] presented a new and efficient method which was based again on the Lloyd's method for Centroidal Voronoi Tessellation computation.

In addition to Voronoi diagrams based mosaicing schemes, there are still some other approaches. Kim and Pellacini [8] have introduced a general framework for creating so-called Jigsaw image mosaics by minimizing an appropriate mosaicing energy function. Another technique for stroke-based rendering that exploits multi-agent systems has been presented by Schlechtweg *et al.* [9]. The algorithm presented in [10] and [11] was based on directional guidelines and

Figure 1. Image mosaics of a picture of panda. Our irregular tiling with photographic tile images composed of patches taken from Chinese ink and wash paintings(left), with patches from Monet's paintings(middle) and irregular tiling result with tile images of stone textures(right).

distance transform and used some known image processing techniques in order to obtain a precise tile placing. Kim *et al.* [12] used mosaic representation for video navigation and they were mainly addressing the problem of packing and appropriate labeling of the given video frames in pre-segmented input images. Recently, Orchard and Kaplan [13] introduced Cut-Out Image Mosaics where arbitrarily shaped image parts were chosen to assemble the final mosaic. As for photo-mosaics, McKean began to create the earliest example of mosaics using photographs in 1994. Silvers [14] worked on the photo-mosaic by matching the target image in tone, texture, shape and color with small photographic images. Finkelstein *et al.* [15] proposed an extension of Silvers' idea which first placed the tile images and then altered their colors to better match the target image. Then Klein *et al.* [16] extended Silver's original idea to videos obtaining a video mosaic which was a two dimensional arrangement of small source video-tiles that suggested a larger video.

Very recently, 3D mosaicing has become a topic of interest. Lai *et al.* [17] showed how to create decorative mosaics on 3D surfaces . Dos Passos and Walter presented a technique to simulate 3D mosaics where the sizes of the individual pieces varied according to the local geometry [18]. They also proposed a method to simulate the *Opus Palladium* effect of 3D mosaic by representing tiles as Voronoi polygons computed from a distribution of points on the surface of the 3D object [19], [20].

The most frequently used regular tiles in image mosaics are rectangles including squares. With equal spacing of regular tiles (Fig.2 *left*) one has to choose much smaller tiles in order to depict features in I well in M, thus making images in tiles difficult to see unless they are enlarged many times. Equal spacing of regular tiles also have the visual periodicity. To avoid such visual periodicity, Pavic *et al.* [21] presented an adaptive tiling method for image mosaics where a bottom-up merging is applied to neighboring non-feature tiles (Fig.2 *middle*). In Fig.2 *right* we include a hand-made glass mosaic by Ulmann, in which the glass is cut into irregular shapes to fit corners in features. There is no literature evidence that such irregular tiling has been simulated in digital image mosaics.

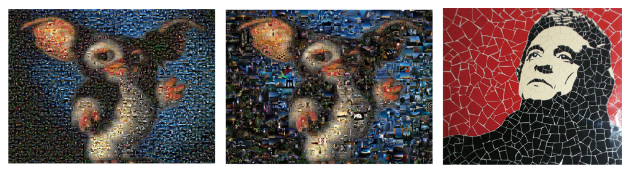

Figure 2. Mosaics with different tiling patterns. Image mosaic with equal spacing tiling (left) and adaptive tiling (middle) [21]. A hand-made glass mosaic (local) by Ulmann (right).

3. System overview

In this work we propose a method to generate image mosaics with irregular tiles that mimic the irregular tiling pattern in Fig.2 *right* using polygon tessellation. Our method introduces images into irregular tiles which distinguishes our work different either from previous irregular tiling approaches in which each tile has just a single bit of color, or from previous image mosaics using just regular rectangles as tiles. The basic architecture of our system is presented in Fig.3. We tessellate the entire image region of I with polygons and then modify the tessellation by the use of the edge points obtained from a feature-based segmentation (Section 4). The tile size in the tessellation can be further altered by users to meet their artistic demands. Once the tiling pattern is obtained, we match the tile regions in I and tile images in T using a 28-dimensional image descriptor to generate the final image mosaic M.

In the following two sections we proceed to detail our

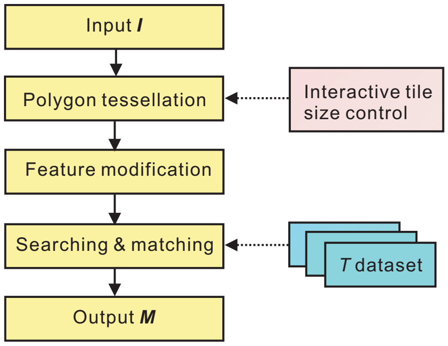

Figure 3. System workflow overview.

method. Section 5 presents some image mosaics generated by our method. We conclude our work in section 6.

4. Irregular tiling by polygon tessellation

This section describes our irregular tiling strategy which involves three steps: (1) tessellating the entire image region of I with polygons, (2) segmenting I into regions depicting object features with the EDISON algorithm [22], and (3) moving some vertices in the polygon tessellation toward the nearest points of region edges.

4.1. Tessellation of polygons

Tessellations in mathematics mean that shapes which tessellate cover the plane without gaps and without overlapping. Tessellations using single triangles or quadrilaterals are not desirable in image mosaics because the resultant tessellation has noticeable repetitions as shown in Fig.4.

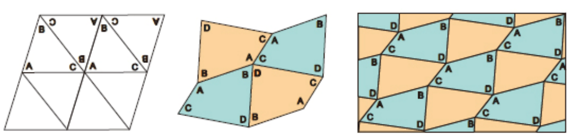

Figure 4. Left: tessellation by triangles. Middle and right: tessellation by quadrilaterals.

We therefore tessellate with five sided polygons (pentagons), because the pentagon can be further divided into a triangle and a quadrilateral, two quadrilaterals, or three triangles, and in these ways the visual periodicity appearing in the triangle and quadrilateral tessellation can be negated.

Unfortunately, not all pentagons tessellate naturally. Since the sum of the interior angles of an n-sided polygon is given by $180° \times (n-2)$, the sum of all the pentagon's interior angles is therefore $540°$. However, this sum of $540°$ is not helpful when trying to fit a number of pentagons around a vertex. In 1918 Reinhardt found five types of pentagon which tessellated properly [23]. Fifty years later Kershner gave three more new types [24]. In 2000 Sugimoto and Ogawa proposed a method for classifying convex pentagons suitable for tessellations and discussed 11 types of pentagon tessellation that was reported in the literature [25]. We choose type 4 (as shown in Fig.5 I) in their classification as a candidate for our initial tessellation because it produces more irregular looking tessellation than others.

Fig.5 *left* shows the process of tessellation with the chosen pentagon. We initially set a=b=$2Sr$, and c=d=Sr, where Sr controls pentagon's size which can be adjusted through user's intervention.

Although a tessellation using this pentagon looks more irregular visually than the tessellations using triangles and quadrilaterals (Fig.4), the visual periodicity induced by repetitions of hexagons is still noticeable in the result (Fig.5 III). To avoid this visual periodicity, we first add small perturbations to the vertices of the pentagon tessellation, then pick up deformed pentagons resulted from addition of perturbations at random and divide them into two or three patches (Fig.5 IV). The resultant tiling pattern has a hand crafted look, as expected.

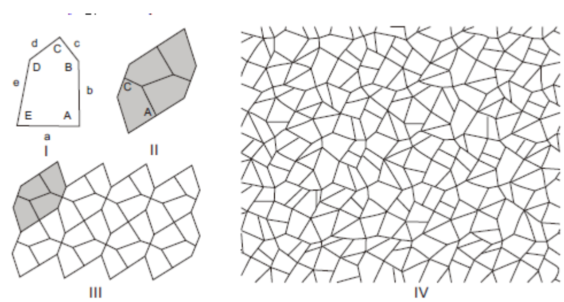

Figure 5. I: Indication of arrangement of angles A, B, C, D and E and edges a, b, c, d and e for the pentagon, where A = C =$90°$, a = b, c = d. II: Tessellation of four pentagons to form a hexagon. III: Tessellating a large region with hexagons through a series of simple translations. IV: The tiling pattern after adding perturbations and performing subdivision.

In the feature regions of the hand-made glass mosaic (Fig.2 *right*), edges of irregular tiles are made to coincide with those of feature regions so that target object features can be better preserved. As our tessellation proceeds independently of target features in I, there is a tendency for the final positions of polygon edges to be drawn away from edges of object features in I, as shown in Fig.6 *left*. If we are to depict feature regions effectively, we need to match polygons vertices in the tessellation more closely with feature edges in I. This requires the segmentation of I into salient feature regions from which feature edges can be obtained, as described next.

4.2. Image segmentation and tessellation modification

Image segmentation is a process in which regions or features sharing similar characteristics are identified and grouped together. Chabrier *et al.* evaluated four major segmentation algorithms and they concluded that the Edison segmentation seemed to be visually the best one [26]. Following Chabrier *et al.* we favor the EDISON algorithm here, and use the code available from [22] to perform our segmentations.

After the EDISON algorithm is applied to *I*, a set of points *Pi* which describes the edge points of segmented regions, is obtained from the edge results provided by EDISON. We then sample every 10th point in the ordered set *Pi* to get a smaller set of edge points *Pj*, as this is sufficient to preserve edge features. This also enables a quicker search between *Pj* and vertices *Vi* in the polygon tessellation. Next, we take a vertex *Vi'* in *Vi* as the center of circle of radius *Sr* and search for all vertices in *Pj* within that circle region, and choose the point *Pj'* in *Pj* nearest to *Vi'* in that circle. Finally we replace the point in *Vi* by the point selected from *Pj*. The process continues until all the entries have been examined. Fig.6 *right* illustrates typical results of tessellation modification.

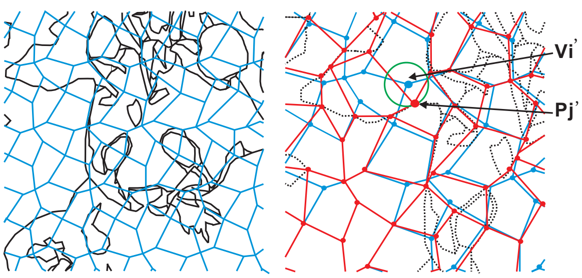

Figure 6. Left: Edges in black of segmented regions of the picture of panda and irregular tiling in blue and modified tessellation in red. Right: the local of tessellation modification with *Pj'* and *Vi'* indicated.

5. Tile matching and color correction

In image mosaics there are many possible choices for T, photographic images, paintings, icons, textures, and even abstract patterns. The number of images in T involved in existing image mosaic systems may range from dozens to a few thousand, and even to a million [21]. Both larger and smaller T datasets have advantages and disadvantages from different perspectives. A small T may be made up so that the content in T may have some associations with the target image *I* for artistic and commercial purposes. But for a very large T it is almost impossible. Then a smaller T contains a limited range of luminance and so may not cover the range of luminance in *I*. So color correction is needed for a better visual matching. Thirdly, a large T requires more time to search and match than a smaller T does. Whatever kind of T is used, matching between tiles in *I* and images in T must be performed during mosaic generation.

5.1. Matching

In regard to matching, any method proposed in the field of image retrieval can be utilized in mosaic generation. It is beyond the scope of this paper to review all techniques for image retrieval. Our matching mechanism is primarily based on a very practical method given in [27], in which each tile image is partitioned into a 3×3 grid and for each grid cell the average RGB color is computed. This leads to a 27-dimensional image descriptor. However this RGB color based image descriptor does not take the region roughness into account. For instance, a tile image with a low roughness value but similar average colors to the tile region in *I* may be selected from T to paint a tile region with a high roughness value. Such mismatches is visually not desirable in image mosaics.

Since entropy is the metric which is most helpful in determining the roughness of an image, we introduce an additional dimension, the tile entropy *H*, to measure the region roughness in the image descriptor in order to obtain a better matching of both color and roughness. The matching between the tile in I and image in T is then achieved by minimizing the quadratic functional of the following 28-dimensional image descriptor (ImD):

$$ImD(C,H) = \alpha(\frac{H_S - H_T}{H_{max}})^2 + \beta \left\{ \frac{1}{3 \times \mathbf{K}(V_{I\cap T})} \sum_{C\in\{R,G,B\}} \sum_{i\,\in V_{I\cap T}} (\frac{C_{S,i} - C_{T,i}}{C_{max}})^2 \right\} \quad (1)$$

where H_S and H_T are the entropies of source and target images, respectively. $H_{max} = 8$ is the maximum entropy value we may obtain from ordinary images. $V_{I\cap T}$ contains the index pairs of the matched vertices in $I \cap T$, $\mathbf{K}(V_{I\cap T})$ is the cardinality of $V_{I\cap T}$ (=9 here), and $C_{max} = 255$ is the maximum value for each RGB component. α and β are factors which weight the region entropy relative to the 27-dimensional RGB color. From experiments we set $\alpha = 0.4$ and $\beta = 0.6$ as default to keep a balance between the RGB color and the region roughness for matching. Users may further tune them via our system's interface to explore other possibilities. To show the effectiveness of entropy in the descriptor, we include some images with pure green colors and pure black colors into *T*. And we can see the result using only RGB image descriptor in Fig.7 *middle*, where the green background with grasses and bamboos is covered by the tile images with plain color variations and low roughness and the body is covered by pure black tiles. These smooth images are selected because of similarity between the average color of them and the green background. In contrast, our image descriptor takes roughness into account and then picks up the images with higher roughness instead of those smooth green images and black images to cover the green background and the body in *I*, as shown in Fig.7 *right*.

In our system all tile images are square in shape. However the irregular tile region and square tile image create a mismatch condition. Our solution is first to calculate the bounding box for each irregular tile and then use the sub-image of *I* to match the tile image in *T*. In order to preserve the aspect ratio when copied into the tile region in *M*, we first place a virtual square (tile) over the irregular tile in *I*, the length of the square being equal to the maxima between the width and height of the bounding box. Next, we scale the matched tile image to fit the virtual square and copy the tile image onto the virtual square. Finally, we take the irregular region in *I* as a binary mask and copy pixels in the virtual square in *I* into the tile region in *M*. We note that parts of the tile images on the outsides of irregular tiles are cut out, thus parts of the tile images are missing in the resultant mosaics. However this is not a serious problem because the salient information is often in the central area on the tile image and our irregular tiling scheme retains that.

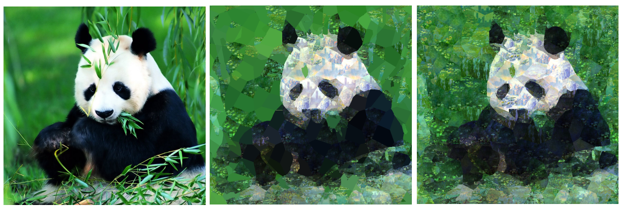

Figure 7. Source image(left), a mosaic image generated with RGB image descriptor (middle) and a mosaic generated with our image descriptor (right).

5.2. Color correction

In our system we adopt a small *T* which contains 50-100 images. Adoption of a small tile dataset on the other hand requires color correction on tile images after matching. We choose the algorithm given in [15] to correct colors in tile images because it is sufficient for our purposes and easy to compute. The correction rule is given in the pseudocode below but more detail is given in [15]. The function *position()* transforms the position indices or coordinates of $T_y \in T$ to the same index space I uses around $V_x \in V$, the common vertices in $I \cap T$, where T_y is the current tile and V_x a vertex in I to which T_y is aligned. We include functions Γ_1, Γ_2 to vary the input colors off I and the tile respectively, non-linearly, e.g by brightness or position relative to the center of T_y.

A = {(i,j)$|X_{i,j} \in$ (position(T_y, V_x)) $\wedge\, T_y \in T \wedge V_x \in V$ }
for all indices k$\in$ A **do** a = a + Γ_1(I[k])
for all colors C $\in T_y$ **do** {C_average = C_average + Γ_2(C) }
f = a / C_average
for all colors C $\in T_y$ **do** { C = C * f }

Table 1. Color correction model.

6. Results

We present some results generated by our irregular tiling scheme and small *T* datasets. In the first example we take a picture of panda as *I* and collect patches from Chinese paintings and Monet's paintings to construct the relevant datasets, and generate two image mosaics as shown in Fig.1 *left* and *middle*. While Fig. 1 *right* shows a mosaic image with irregular tiles composed of stone textures. The gaps between tiles are rendered with lines of varying width to make the grout look more natural. The color of the grout is gray which is adaptively varied in brightness inversely with the average luminance of its neighboring tiles.

In Fig.8 we include two previous digital mosaics with irregular tiling for comparison. First, the Voronoi diagram based approaches produce tiles with similar shapes, while our method generates tiles with shapes that change adaptively according to features of the target image. Also our method use less tiles in the resultant mosaics than those used in previous mosaics of irregular tiles.

Fig.9 shows the mosaic effects of two hands in Michelangelo's painting, from which we can see that Voronoi cells tend to produce zigzag boundaries along the hand contour(Fig.9 *middle*), while our feature based segmentation scheme tends to preserve the boundary shapes of the two hands (Fig. 9 *right*). In the following figures we show some examples of portrait image mosaics generated by our method. Such portrait mosaics, to our knowledge, are not presented with previous irregular tiling schemes including both Vornoi diagram based methods or the Jigsaw image mosaics in which the tiles are some chosen objects. Also we introduce patches taken from paintings and texture images into our tiles to produce varying mosaic effects that are different from those produced by Voronoi diagrams.

Fig.10 *left* is a Buddha statue which is used as *I*, the dataset is composed of patches from line drawings of Buddhism mdra (mystic positions of the hand), as shown in Fig.10 *middle*. Fig.10 *right* shows the generated mosaic with stone texture patches.

In Fig.11 we present another example in which the target image is Marilyn Monroe's portrait in gray-scale, while the datasets are constructed by patches taken from Monet's paintings (Fig.11 *middle* and ceramic textures in Fig. 11 *right*. In this example, we use color tile images to depict a gray-scale input image which brings a unique quality to the resultant image mosaics.

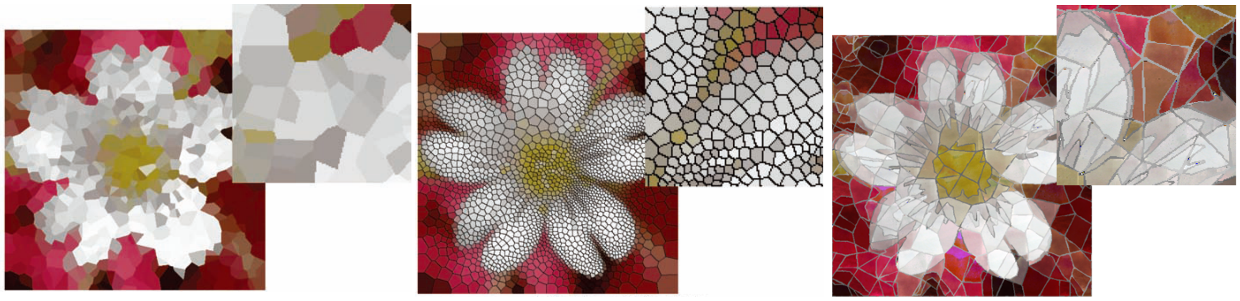

Figure 8. From left to right: Examples of digital mosaics with irregular tiling by the method given in [2], [4] and ours.

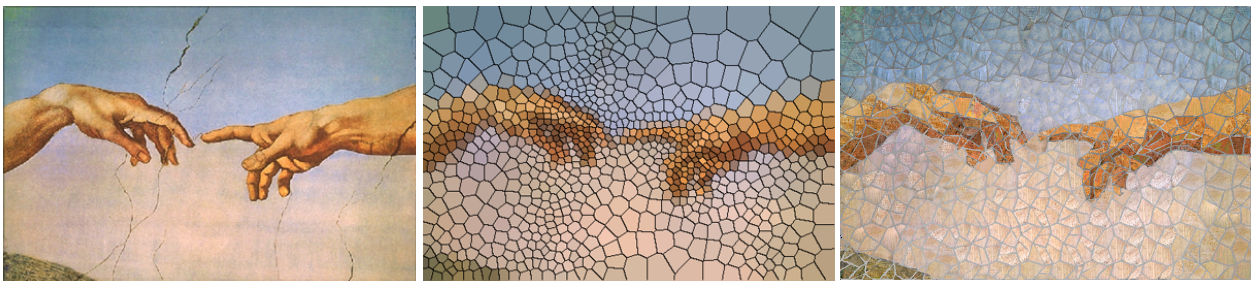

Figure 9. Left: the input image, middle: result given in [4], right: our result.

Figure 10. Image mosaics of Buddha

Fig.12 takes Barack Hussein Obama's portrait as input. We use patches taken from oil paintings and glass textures to generate two image mosaics, as shown in Fig.12 *middle* and *right*.

In Fig.13 we show the last example of our image mosaics, Venus. Fig.13 *middle* is generated with a dataset T constructed by collecting Chinese fine flower-and-bird paintings and Fig.13 *right* with stone textures.

Table.2 shows the summary of image size, tile number and running time required for previous examples.

	Image size	Tile number	Running time
Panda	780×768	558	4141ms
Buddha	800×974	731	4032ms
Monroe	600×670	367	2603ms
Obama	800×924	846	4265ms
Venus	700×812	534	3980ms

Table 2. Summary of image size, tile number and running time required for previous examples.

7. Conclusion

In this paper we have presented a system capable of generating image mosaics with irregular tiling based on

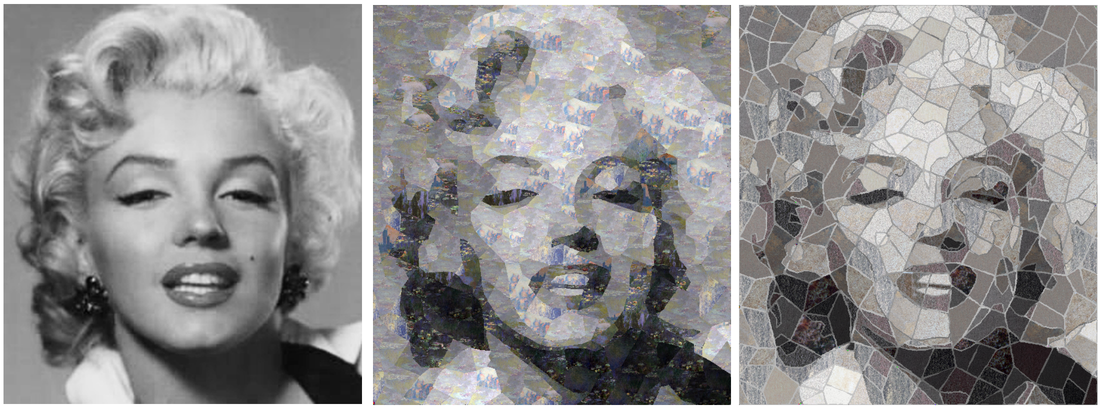

Figure 11. Image mosaics of Marilyn Monroe

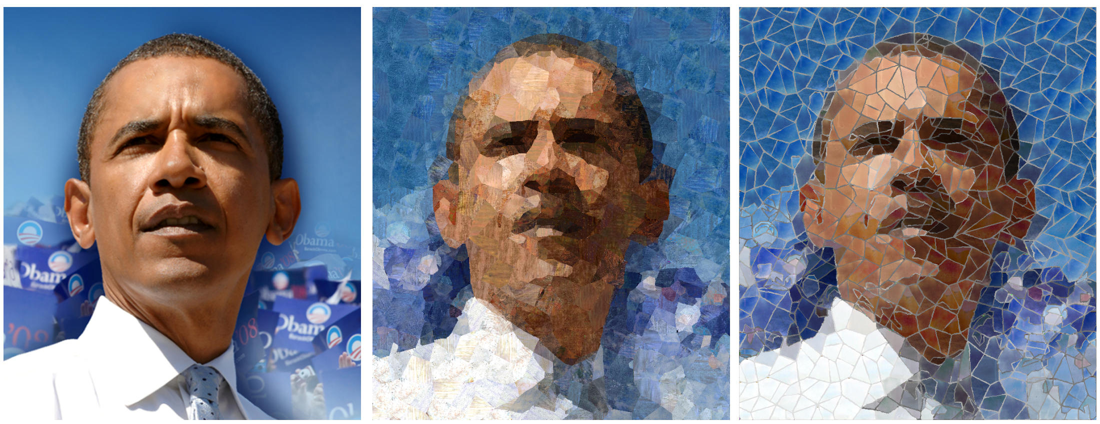

Figure 12. Image mosaics of Obama

Figure 13. Image mosaics of Venus

polygon tessellation, using the smaller tile image datasets. The key issue associated with image mosaic generation using small T dataset is the feature size match between the tile images and input images. In our current work, the minimal tile size is decided by user interaction. A future work is to find a method to measure the feature size for images of arbitrary objects, so that the tile size can be determined automatically.

In our current image mosaic system, the tile image datasets are constructed manually according to the associations between the input images and tile images. It would be desirable that the tile image dataset can be constructed automatically or semi-automatically. This requires semantic recognition of some fixed class of objects in the given images, which is also a topic for future research.

Acknowledgment

This work is supported by the State Key Program of National Natural Science Foundation of China(No.60933007), and the Key Technologies R&D Program of China(No.2007BAH11B02).

References

[1] Battiato, S., Blasi, G. D., Farinella, G., and Gallo, G., Digital mosaic frameworks - an overview, *Computer Graphics Forum 26*, John Wiley & Sons, 4(2007), pp. 794-812.

[2] Haeberli, P., Paint by Numbers: Abstract Image Representation, *In SIGGRAPH'90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, ACM New York, NY, USA, 1990, pp. 207-214.

[3] Dobashi. J., Haga. T., Johan. H., Nishita. T., A Method for Creating Mosaic Images Using Voronoi Diagrams, *In Proceedings of Eurographics 2002 Short Presentations*, 2002, pp. 341-348.

[4] Faustino, G. M., De Figueiredo, L. H., Simple Adaptive Mosaic Effects, *In: Proceedings of SIBGRAPI 2005*, IEEE Computer Society Washington, DC, USA, 2005, pp.315-322.

[5] Hausner, A., Simulating decorative mosaics, *In SIGGRAPH'01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM New York, NY, USA, 2001, pp. 573-580.

[6] Elber, E., Wolberg, G., Rendering Traditional Mosaics, *The Visual Computer 19*, Springer, 1(2003), pp. 67-78.

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

[8] Kim, J., and Pellacini, F., Jigsaw image mosaics, *In SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM New York, NY, USA, 2002, pp. 657-664.

[9] Schlechtweg, S., Germer, T., Strothottet, T., RenderBots - Multi-Agent Systems for Direct Image Generation, *Computer Graphics Forum 24*, John Wiley & Sons, 2(2005), pp. 137-148.

[10] Blasi, G. D., Gallo, G., and Petralia, M., Puzzle image mosaic, *In Proc. of IASTED/VIIP2005*, vol. 27, 379-424/623-656.

[11] Battiato, S., Blasi, G. D., Farinella, G., and Gallo, G., A novel technique for opus vermiculatum mosaic rendering, *In Proceedings of the 14th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG 2006)*, 2006, pp. 133-140.

[12] Kim, K., Essa, I., and Abowd, G., Interactive mosaic generation for video navigation, *In MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia*, ACM New York, NY, USA, 2006, pp. 688-658.

[13] Orchard, J., and Kaplan, C., Cut-out image mosaics, *In NPAR '08: Proceedings of the 6th internatioanl symposium on Non-photorealistic animation and rendering*, ACM New York, NY, USA, 2008, pp. 79-87.

[14] Silvers, R., *Photomosaics*, Henry Holt and Co., Inc. New York, NY, USA, 1997

[15] Finkelstein, A., and Range, M., Image mosaics, *In EP'98/RIDT '98*, Springer,1998, pp. 11-22.

[16] Klein, A., Grant, T., Finkelstein, A., and Cohen, M., Video mosaics, *In NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, ACM New York, NY, USA, 2002, pp. 21-28.

[17] Lai, Y.-K., Hu, S.-M., and Martinn, R., Surface mosaics, *The Visual Computer 22*, Springer, 9-11(2006), pp. 604-611.

[18] Dos Passos, V. A. and Walter, M., 3D mosaics with variable-sized tiles, *The Visual Computer 24*, Springer, 7-9(2008), pp.617-623.

[19] Dos Passos, V.A. and Walter, M., Simulation and Rendering of Opus Palladium 3D Mosaics, *in XXI Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2008)*, IEEE Computer Society Washington, DC, USA, 2008, pp. 263-269.

[20] Dos Passos, V.A. and Walter, M., Opus Palladium and mixed styles, *The Visual Computer 25*, Springer, 10(2009), pp.939-946.

[21] Pavic, D., Ceumern, U., and Kobbelt, L., Gizmos: Genuine image mosaics with adaptive tiling. *Computer Graphics Forum 28*, John Wiley & Sons, 8(2009), pp. 2244-2254.

[22] EDISON, 2003, Code for the edge detection and image segmentation system. http://coewww.rutgers.edu/riul/researchcode/EDISON/

[23] Reinhardt, K., *Uber die Zerlegung der Ebene in Polygone*, R. Noske, Borna and Leipzig, 1918.

[24] Kershner, R., On paving the plane, *The American Mathematical Monthly 75*, Mathematical Association of America, 4(1968), pp.839-844.

[25] Sugimoto, T., and Ogawa, T., Tiling problem of convex pentagon, *Forma 15*, Japan, 1(2000), pp. 75-79.

[26] Chabrier, S., Rosenberger, C., and Emile, B., Segmetation evaluation by fusion with a genetic algorithm, *In the 13th European Signal Processing Conference(EUSIPCO 2005)*, 2005.

[27] Blasi, G. D., Gallo, G., and Petralia, M., Smart ideas for photomosaic rendering, *In Proceedings of Eurographics Italian Chapter Conference*,Catania, Italy, 2006, pp. 267-272.