Mosaic: Sketch-Based Interface for Creating Digital Decorative Mosaics

Rinat Abdrashitov ‡
University of Toronto

Emilie Guy *
Telecom ParisTech,
CNRS LTCI, Institut Mines Telecom

JiaXian Yao †
University of California,
Berkeley

Karan Singh ‡
University of Toronto

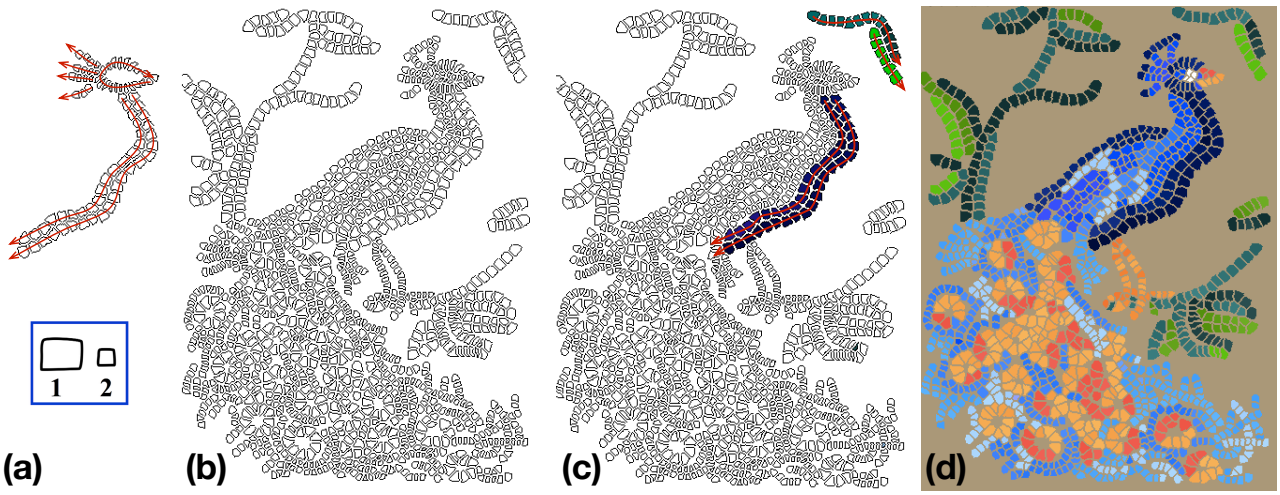


Figure 1: *Creative workflow in Mosaic: (a) Two mosaic tile pieces of different size and shape are sketched. The first and second tiles are laid out along sketched paths to create part of the body and head respectively, the pieces placed and deformed automatically, to maintain a uniform grout. (b) More tiles are cloned to complete the mosaic. (c) Sketch strokes are then used to color the tiles and the grout area. (d) A peacock, created in 40 min.*

Abstract

Mosaic is a sketch-based system that simplifies and automates the creation of digital decorative mosaics from scratch. The creation of each tile piece of unique shape, color and orientation, in a complex mosaic is a tedious process. Our core contribution is two-fold: first, we present a new tile growing algorithm, that balances the shape and placement of tiles with need for uniform grout; second, we develop a suite of sketch-based tools on top of this algorithm to create and clone tiles and tile patterns along sketched paths, and color them efficiently. A user evaluation shows that our system makes the creation of mosaics fast and accessible to a broad audience.

CR Categories: I.3.3 [Computer Graphics]: Curve Generation

Keywords: sketching, curves, mosaic, decoration

1 Introduction

Mosaic is the art of creating pictures or patterns by arranging small colored pieces of material, such as stone, tile, glass or even paper, together. Mimicking the alignment and packing of elements in nature, mosaics have applications in architecture, design, furniture, jewellery and children's craft [Kelly 2004]. One of the most ancient and inventive forms of surface decoration, mosaic design is equally ubiquitous today. They have been found across the entire areas of the Roman Empire, in the Middle East and Western Asia, exhibiting and enormous range of genres and styles [Massey and Slater 1999]. Historically, mosaics were popularized by the inexpensive availability of uniform tiled pieces as well as scrap and found material. At present, increasingly common 2D routing devices such as laser cutters have further eased the ability to fabricate custom pieces for use in mosaic art.

Creating a mosaic both physically, or as a digital image, from scratch can be a painstaking process. Standard commercial drawing applications like Illustrator, Photoshop, or Autodesk Sketchbook require users to tediously manipulate individual pieces, and do not capture the higher level structure of a mosaic needed to facilitate rapid creation and coloring of large numbers of mosaic pieces.

Our system is inspired by two observations relevant to modern mosaic construction: variability of mosaic pieces and the importance of uniform grout. As observed by practical texts on mosaic design [Kelly 2004], while individual mosaic tiles locally define an image, their arrangement defines the visual flow within the image. While mosaics have traditionally been composed of regular tiles, variations in otherwise regular mosaic pieces can positively impact the aesthetic of an image, as with patterns observed in nature [Turk 1991; Pedersen and Singh 2006]. The space in-between tiles, known as grout, defines a negative space within which the mosaic is embedded. Professional mosaics strive to maintain a uniformly spaced grout throughout the image, as variations in grout draw visual attention away from the mosaic itself. We present an algorithm that grows mosaic pieces to balance the needs of uniform grout and regular tiles.

*emilie.guy@telecom-paristech.fr
†jiaxian.yao@eecs.berkeley.edu
‡(rinat|karan)@dgp.toronto.edu

There is a large body of research on and related to digital mosaic creation [Battiato et al. 2007], focused on a range of styles with monikers such as crystallization, ancient, photo and puzzle mosaics (Figure 2). The majority of these approaches are largely automatic, numerically optimizing a tiling fitness function relative to an input image, with little user interaction. We instead focus on the creative process, allowing users to bring their own style to bear in creating mosaics by interactive sketching on a blank canvas, or tracing over reference imagery.

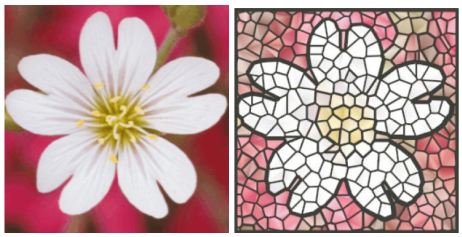


Figure 2: *Mosaic image (right) created from the source image (left) using voronoi diagrams [Dobashi et al. 2002].*

Our system, Mosaic, uses a new tile growing algorithm that simplifies the creation and placement of uniquely shaped mosaic pieces, with uniform grout. Our algorithm iteratively, expands and moves each mosaic tile from an initial centroid location, subject to forces that attempt to preserve tile-shape and repel other tiles to maintain uniform grout. Even a single mosaic piece drawn by the user is enough to tile a creation. The system provides sketch-based tools to create a mosaic accurately piece by piece, as well as to automatically clone and fill regions or paths with multiple pieces. Sketch-based coloring of multiple pieces at once with the same color or creating a color gradients is also supported. Users have full interactive control to impose their own style over the created mosaics.

In a typical workflow, the user first creates a palette of few mosaic pieces by drawing tiles of different shape and size (Figure 1a), and then outlines the main features of the mosaic using a few strokes. These stroke gestures allow the user to quickly clone mosaic pieces to fill space in desired patterns, leaving the tedious aspects of tile packing to our tile growing algorithm (Figure 1b). Using similar stroke gestures the user can also color mosaic pieces to create the final result (Figure 1c, d). Using Mosaic, first-time users are able to create complex and original mosaics within minutes.

Our technical contribution is twofold:

1. A tile growing algorithm for the creation of mosaics with uniform grout.
2. A suite of sketch-based tools that build upon the algorithm, to clone and color mosaic images from scratch.

The remainder of this paper is structured as follows. We discuss NPR approaches to mosaic creation in related work (Section 2), followed by our user interface, tools and system workflow (Section 3). Section 4 details the tile growing algorithm. Finally, we present an evaluation and results of system usage by users of varying artistic skill and experience (Section 5).

2 Related Work

A large body of research in Non-photorealistic rendering addresses the transformation of raster images into a various artistic representations [Pedersen and Singh 2006; Li and Mould 2011], including mosaic of various styles. The bulk of these techniques are automatic. For example, Hausner [2001] presents a method to simulate decorative mosaics that employs centroidal Voronoi diagrams to arrange square tiles and uses an imposed direction field to align tiles with edge features chosen by a user. Voronoi techniques have also been used to automatically preserve both features and color variation of an original image [Dobashi et al. 2002]. These and similar methods [Elber and Wolberg 2003; Di Blasi and Gallo 2005] are not aimed at user interactivity (for an overview, see [Battiato et al. 2007]). Commercial applications like Adobe Photoshop and GIMP are also able to produce mosaics from an input image, but similar to the above research provide users with little interactive control.

Similarly to Mosaic, there are other digital systems that are inspired by the real-world art forms. SandCanvas [Kazi et al. 2011], Project Gustav [Chu et al. 2010], Fluid Paint [Vandoren et al. 2009], DECO [Igarashi 2011] and Holy [Igarashi and Igarashi 2010] are digital systems that deliver various tools to the artist which strongly resemble real-world techniques. Vignette [Kazi et al. 2012] facilitates texture creation in pen-and-ink illustrations. Users draw a fraction of a texture and use gestures to automatically fill regions with the texture or repeat textures along the drawn path. Mosaic uses similar ideas to simplify and automate the creation of mosaic pieces. However, unlike Mosaic, Vignette does not embody the notion of grout or negative space, nor does it facilitate deformation of pieces in order to create unique shapes, it only repeats or rearranges them.

3 Mosaic: Interface and interaction

The main window in Mosaic has three panels (Figure 3). A Draw Panel, which allows users to create mosaic pieces of any shape. A Drop panel has slots for frequently used mosaic pieces. A Clone Panel is the main drawing canvas where users create mosaics by cloning one or more mosaic pieces created in the Draw Panel. Users select from a suite of tools (Figure 4) to perform operations in the Main window.

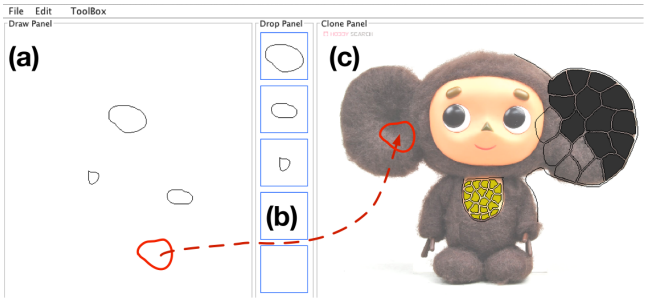


Figure 3: *Main window. (a) Mosaic pieces are created in the Draw Panel. (b) Drop Panel allows to quickly switch between mosaic pieces when cloning using the Guide Clone tool. (c) Mosaics are arranged in the Clone Panel by cloning mosaic pieces.*

Almost all operations in Mosaic are performed using simple stroke or drag gestures. These operations include drawing, selecting, cloning, erasing and coloring mosaic pieces:

Background image tool. Users can load a background image into the Clone Panel as a visual reference, with variable transparency, to aid in mosaic creation (Figure 3c).

Pencil tool. Users sketch strokes in the Draw and Clone Panels. In the Draw Panel users can draw curves to be used as individual mosaic pieces or small arrangements of a set of mosaic pieces. In the Clone Panel users sketch strokes (with user controlled smoothness), that act as *boundary strokes* that contain the growth of mosaic tiles to one side or the other of the curve.

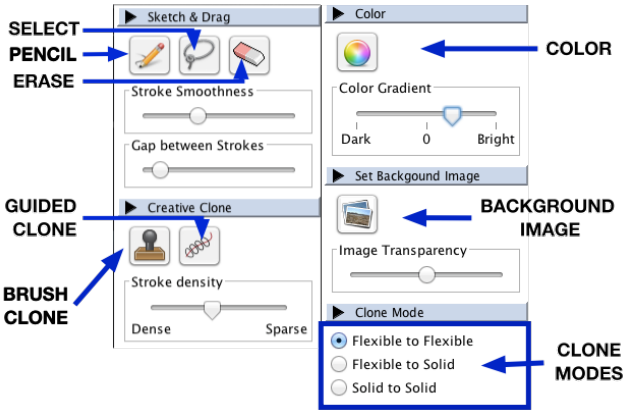


Figure 4: *Toolbox.*

Select tool. User performs selection by stroking over one or multiple mosaic pieces in the Draw Panel. The Drop Panel allows users to switch between frequently used pieces directly without employing the select tool.

Erase tool. User can erase one or multiple mosaic pieces by stroking over them in any panel.

Clone tool. When precision is required, users can clone mosaic pieces one by one. Pieces selected in the Draw Panel, can be dragged and positioned to the desired location in the Clone Panel as illustrated in Figure 3 (mosaic piece in red). The rigidity of each mosaic piece can be defined, so that a range of deformable behaviours can be accommodated, in cases where the new piece overlaps existing pieces (Figure 5).

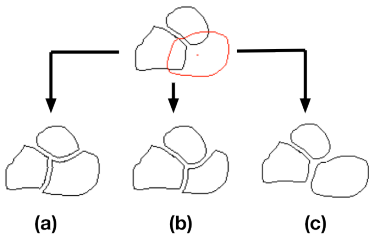


Figure 5: *Three clone modes. Piece being added is highlighted in red. (a) All pieces are deformed. (b) Only added piece is deformed. (c) None of the pieces are deformed. Added piece is moved to avoid collision.*

Guide Clone tool. Most mosaics use the lines of grout in-between mosaic tiles to define flow-lines within an image. It is thus natural for users to define paths along which mosaic pieces are laid out by sketched strokes. Users can thus simply select one or more mosaic pieces in the Draw Panel (Figure 6a), and sketch a guide line (Figure 6b) in the Clone Panel. The sketched guide line is automatically populated with selected pieces, aligned along the stroke, that fill-in using our tile growing algorithm (Figure 6c). Users can dynamically control the tile density of pieces placed by the tool (Figure 6d, e and Figure 7).

Brush Clone tool. Users may create local patterns and arrangements of multiple mosaic pieces that they may wish to selectively clone. Users select a starting point in the Draw Panel (Figure 8a), and then brush over the Clone Panel. Brush movements in the Clone Panel are then synchronized with the Draw Panel and mosaic pieces that are brushed over are cloned (Figure 8b). In our implementation,

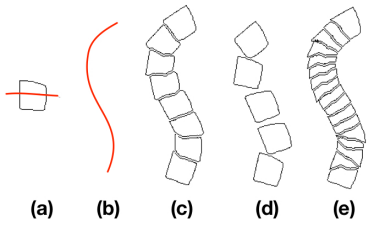


Figure 6: *Guide clone tool. (a) Select a piece to be cloned. (b) Stroke a guide line. (c, d, e) Pieces are cloned and deformed along the guide line with different density.*

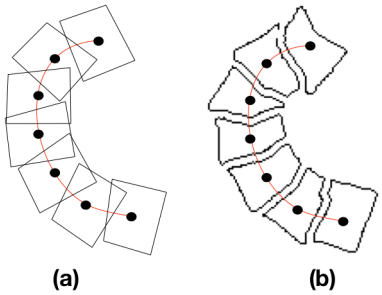


Figure 7: *(a) Pieces are cloned along the stroke and rotated. (b) Tile growing algorithm is applied to deform the pieces.*

if the user brushes over more than 25% of a piece, then the entire piece is cloned (Figure 8c).

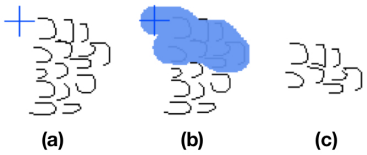


Figure 8: *Brush Clone tool. (a) Set a start point. (b) Brush over pieces. (c) Selected pieces are cloned.*

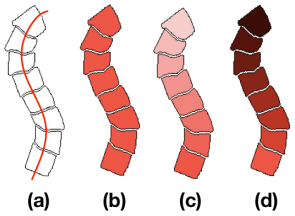


Figure 9: *Color tool with gradient effect. (a) Stroke over the pieces (b) Chosen pieces are colored. (c, d) Gradient effect is an option.*

Color tool. Mosaic tiles can be colored in all three panels. Users color pieces by simply stroking over them, filling them with a chosen solid color or color gradient (Figure 9). Grout color can also be set by stoking anywhere in the background.

A typical workflow in Mosaic starts by drawing a few boundary strokes (optionally over a background image) to delimit the mosaic. After that, mosaic pieces are drawn and selected from the Drop Panel. The mosaic itself can then be defined with precision using the Clone tool, or quickly filled along paths using the Guide Clone

tool or in regions using the Brush Clone tool. The tile growing algorithm interactively ensures that the mosaic pieces grow to respect the boundary curves and trade-off between tile rigidity, placement and grout uniformity. The Color tool can be used at any point during this workflow to color the mosaic.

4 Tile Growing Algorithm

All mosaic pieces are represented as 2D poly-lines. In our implementation we rasterize them at high resolution into connected set of 2D pixels (Figure 10). This representation forms the input to our tile growing algorithm.

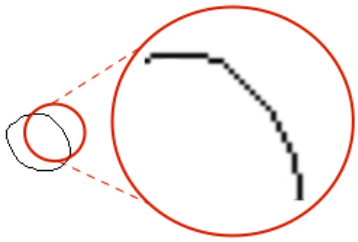


Figure 10: *Zoomed mosaic piece. Points are tightly packed.*

Our tile growing algorithm is an iterative simulation where points corresponding to mosaic pieces and boundary curves, evolve under various forces at each step. Assume that some tile has N points, with orginal positions o_i, and evolving positions p_i $(1 \leq i \leq N)$. We support three forces acting upon the evolving points :
-a *growth force* G that pushes points outwards from the tile centroid $c = (\sum_i(o_i))/N$ towards their original location on the tile shape. $G_i = g * (o_i - c)$ where g is a fractional constant. After k iterations the shape is a $k * g$ scaled version of the tile around its centroid c.
-a *repulsion force* R between nearby points that is responsible for maintaining uniform grout, preventing interpenetration between mosaic tiles, and between tiles and boundary curves. $R_i = \sum_j vdw(||q_j - p_i||) * (q_j - p_i)$, where vdw is a van der Waals force between p_i and points on other tiles and boundaries q_j, within a radius of influence d. The force vdw is exponentially negative or repulsive for points closer than the desired grout thickness t, and positive or attractive for point distances between (t, d) dropping to zero at distance d and beyond [Pedersen and Singh 2006].
-a *shape force* S that rigidly transforms all points of a tile (at its currently grown scale) to best fit the tile's current deformed shape. In other words, we compute the rotation A and translation T of the tile at the current or kth iteration scale (points $v_i = c + k * g(o_i - c)$), such that $||(Av_i + T) - p_i||^2$ is minimized. The solution of A and T to this expression due to Horn [Horn 1987], is well known [Breslav et al. 2007], the translation T for instance being simply $(\sum_i(p_i))/N - c$. The shape force S_i is simply a linear blend $S_i = p_i + rigid * ((Av_i + T) - p_i)$, where $rigid \in [0, 1]$ is a shape rigidity parameter.

Growth begins by shrinking all points to the centroid of their respective tiles (Figure 11a). Assume that tile has N points, the position of each point, p_i $(1 \leq i \leq N)$, is updated by adding the three forces. At every iteration, the point p_i is updated to $p_i + G_i + R_i + S_i$, growing outwards subject to G (Figure 11b,c), repelling nearby points according to R and maintaining a rigid or deformable shape based on S and the $rigid$ prameter. To keep the simulation stable, iteration stops and point locations are permanently set (points highlighted in green in Figure 11d) once the user desired grout thickness is achieved or a point grows to its original location (we do not scale tiles past their given size). Finally, for each piece we perform a linear interpolation to fill any missing points that result from tile deformation.

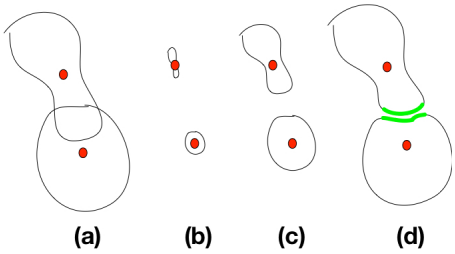


Figure 11: *Tile growing. (a) Illustrates initial state when pieces are added. (b, c) Pieces are then reduced to their centroids and start growing incrementally. (d) If collision is detected for some points, their growth is terminated.*

5 User Evaluation and Results

Four people were asked to participate in our evaluation (2 males, 2 females, age range 22-26 years old, 2 with good drawing skills). All had some experience with non-professional drawing applications. A 9 min video tutorial explaining features and user interface of the application, the software (written in Java) and a questionnaire was sent to them. Users were either using mouse and keyboard or a pen tablet device to draw mosaics.

Figures 12, 13, 14 showcase some of the results. On average, it took 30 minutes to complete the images. Participants' overall impression was very positive. They mentioned that Mosaic was easy to learn (4.25 on a scale of 1 (hard) to 5 (easy)). Participants found that using same stroke-based interactions for creating, deleting and coloring of mosaic pieces made the drawing process pleasant and fast (4 on a scale of 1 (unpleasant) to 5 (pleasant)). They indicated that out of all mosaic cloning tools available they mostly used Guide Clone tool. Interestingly, their strokes are not always evident since our tile growing algorithm uniformly fills in space when the tile density is high enough. Only 1-4 pieces were created and then cloned to complete the images. They were also satisfied with the final results (4.5 on scale of 1 (not satisfied) to 5 (satisfied)). Some mentioned that using our system had motivated them to create mosaics out of real materials.

Some suggestions were made on how to improve the application: bigger canvas, zoom in/out in order to add fine details, multitouch version for a tablet device. One user found the Clone tool to be relatively difficult to understand.

6 Limitations

Limitation of our tile growing algorithm is that points are tightly packed and have 2D interger coordinates (might cause rounding errors). Due to iterativity of our algorithm, those factors might lead to jaggy appearances of some tiles. Jaggedness can be avoided by increasing number of iterations, reducing initial number of points or applying point reduction algorithm, like Ramer Douglas Peucker algorithm, after mosaics are finished. In addition, smoothing and antialiasing can be applied to get more pleasing results.

7 Conclusion and Future Work

We have presented a sketch-based Mosaic application that streamlines and simplifies the often long and painstaking mosaic creation process, using a tile growing algorithm and stroke and brush based tile cloning operations. This application gives users the power to create beautiful mosaic images with only a few simple strokes and shapes. The application is accessible to a broad audience and even beginners can create artistic mosaics that carry their individual style. Our approach also has the potential to create animated mosaics [Smith et al. 2005], since the construction history of guide strokes and the iterative nature of the tile growing algorithm are amenable to handling temporal coherence, and this is subject to future work.

Acknowledgements

We would like to thank ZhengZheng Ye, Heisei Cantero, Yiyan Zhu and Jason Shum for participating in the user study and submitting their beautiful mosaics. Also we would like to thank Bruno De Araujo for his critique on the early draft of the paper.

References

BATTIATO, S., DI BLASI, G., FARINELLA, G. M., AND GALLO, G. 2007. Digital mosaic frameworks - an overview. *Computer Graphics Forum 26*, 4, 794–812.

BRESLAV, S., SZERSZEN, K., MARKOSIAN, L., BARLA, P., AND THOLLOT, J. 2007. Dynamic 2d patterns for shading 3d scenes. In *ACM SIGGRAPH 2007 Papers*, ACM, New York, NY, USA, SIGGRAPH '07.

CHU, N., BAXTER, W., WEI, L.-Y., AND GOVINDARAJU, N. 2010. Detail-preserving paint modeling for 3d brushes. In *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering*, ACM, New York, NY, USA, NPAR '10, 27–34.

DI BLASI, G., AND GALLO, G. 2005. Artificial mosaics. *The Visual Computer 21*, 6, 373–383.

DOBASHI, Y., HAGA, T., JOHAN, H., AND NISHITA, T. 2002. A method for creating mosaic images using voronoi diagrams. In *Proceedings of Eurographics*, vol. 2, 341–348.

ELBER, G., AND WOLBERG, G. 2003. Rendering traditional mosaics. *The Visual Computer 19*, 1, 67–78.

HAUSNER, A. 2001. Simulating decorative mosaics. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York, NY, USA, SIGGRAPH '01, 573–580.

HORN, B. K. P. 1987. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A 4*, 4, 629–642.

IGARASHI, Y., AND IGARASHI, T. 2010. Holly: A drawing editor for designing stencils. *Computer Graphics and Applications, IEEE 30*, 4 (July), 8–14.

IGARASHI, Y. 2011. Deco: A design editor for rhinestone decorations. *Computer Graphics and Applications, IEEE 31*, 5 (September), 90–94.

KAZI, R. H., CHUA, K. C., ZHAO, S., DAVIS, R., AND LOW, K.-L. 2011. Sandcanvas: A multi-touch art medium inspired by sand animation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, New York, NY, USA, CHI '11, 1283–1292.

KAZI, R. H., IGARASHI, T., ZHAO, S., AND DAVIS, R. 2012. Vignette: Interactive texture design and manipulation with freeform gestures for pen-and-ink illustration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, New York, NY, USA, CHI '12, 1727–1736.

KELLY, S. 2004. *The Complete Mosaic Handbook: Projects, Techniques, Designs*. Firefly Books.

LI, H., AND MOULD, D. 2011. Artistic tessellations by growing curves. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Non-Photorealistic Animation and Rendering*, ACM, New York, NY, USA, NPAR '11, 125–134.

MASSEY, P., AND SLATER, A. 1999. *Beginner's Guide to Mosaic*. Search Press, 4–5.

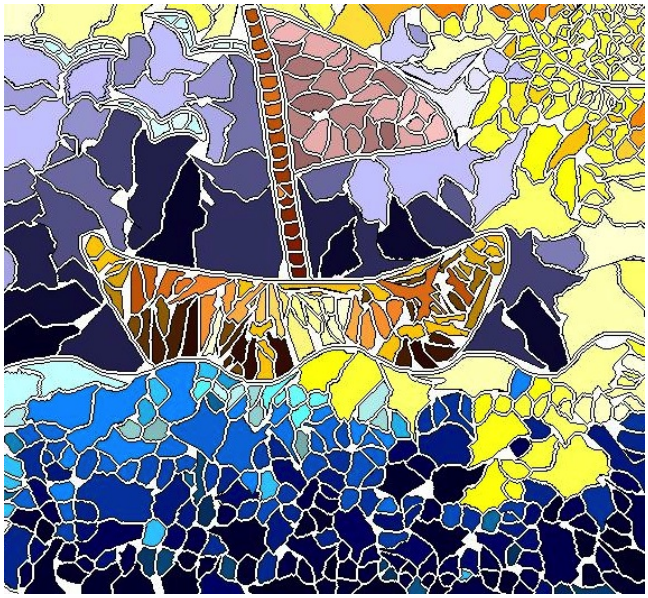
PEDERSEN, H., AND SINGH, K. 2006. Organic labyrinths and mazes. In *Proceedings of the 4th International Symposium on Non-photorealistic Animation and Rendering*, ACM, New York, NY, USA, NPAR '06, 79–86.

SMITH, K., LIU, Y., AND KLEIN, A. 2005. Animosaics. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, SCA '05, 201–208.

TURK, G. 1991. Generating textures on arbitrary surfaces using reaction-diffusion. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York, NY, USA, SIGGRAPH '91, 289–298.

VANDOREN, P., CLAESEN, L., VAN LAERHOVEN, T., TAELMAN, J., RAYMAEKERS, C., FLERACKERS, E., AND VAN REETH, F. 2009. Fluidpaint: An interactive digital painting system using real wet brushes. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ACM, New York, NY, USA, ITS '09, 53–56.

Figure 12: *Pocahontas - 20 min.*



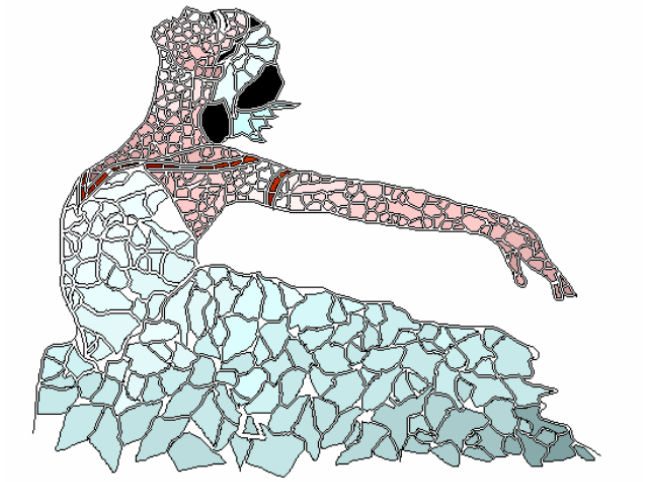
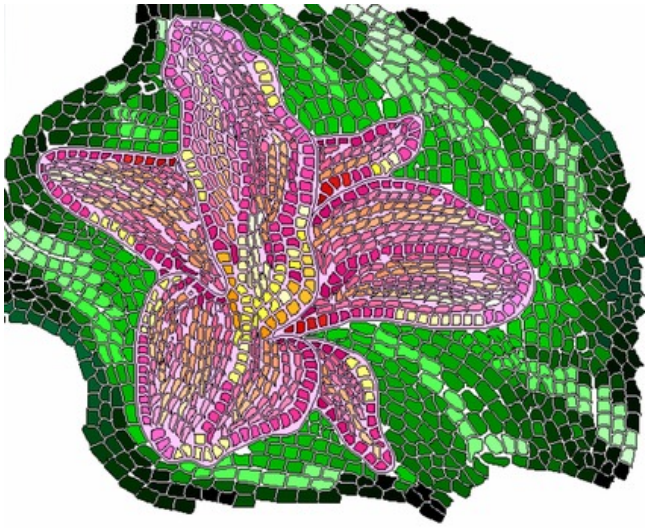
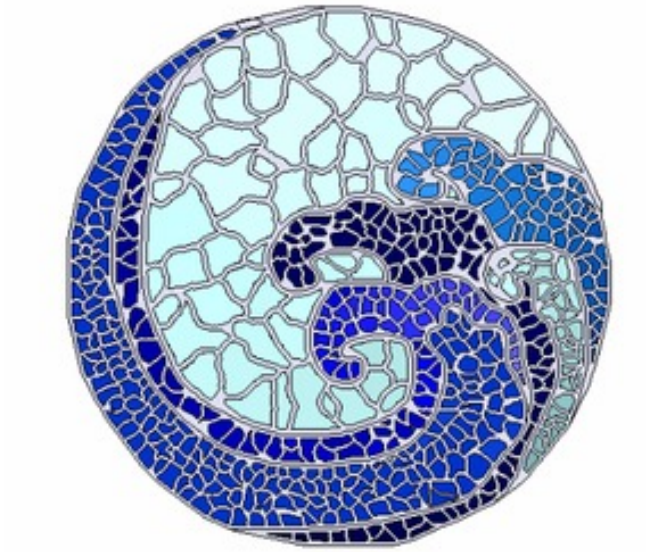


Figure 13: *Boat - 25 min., Ballerina - 40 min., Starry Night - 30 min.*





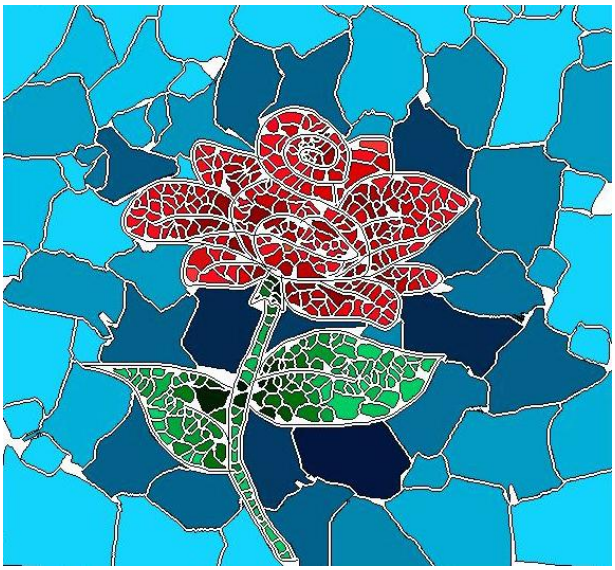


Figure 14: *Lily - 40 min., Wave - 20 min., Rose - 30 min.*