

Grid vs. Arbitrary Placement of Tiles for Generating Animated Photomosaics

Gayan Wijesinghe, Shahrul Badariah Mat Sah and Vic Ciesielski

Abstract—A traditional photomosaic is a still image where a larger picture is created by selectively arranging small picture tiles on a blank, gridded canvas. We show interesting and engaging animations can be generated from an evolutionary search for the final photomosaic image. We then investigate two different tile placement strategies for generating the animations. In the first strategy tiles can only be placed in fixed cells in a 2 dimensional grid and it is not possible for tiles to overlap. This strategy is implemented with a genetic algorithm. In the second strategy, which is implemented using genetic programming, the tiles can be placed in any position and at an arbitrary rotation. It is possible for one tile to be placed on top of another so a method for dealing with overlap is needed. We have investigated three methods for dealing with overlap. The second strategy generates more engaging animations but at considerably increased computational cost. We conclude that evolutionary search can be used to produce very engaging animations in which a target image gradually emerges from an initial random collection of tiles.

I. INTRODUCTION

PHOTOMOSAICS have been a recognised artform for about 10 years. A photomosaic is a large image composed of a number of much smaller images called tiles. A photomosaic is interesting because, when viewed up close, the detail of each tile attracts the viewer's attention, but when viewed from afar, the combination of the tiles reveals a larger image that is the subject of the photomosaic. The interplay between the subject and the tiles provides scope for creative expression. An example of a famous photomosaic is the portrait of President George Bush made of faces of soldiers killed in Iraq [1]. Traditionally photomosaics are still art works designed to be printed and viewed on paper. In contrast, our work is dynamic and designed to be viewed on a motion screen. The interesting and engaging dynamics are captured from an evolutionary search.

As evolutionary searches arrive at optimal solutions gradually, they provide the possibility of visualizing the intermediate solutions as frames of a movie. This approach has generated an artwork (Figure 1) shown at a recent international art exhibition [2], [3]. Here the canvas is treated as a grid and only one tile can be placed in a cell of the grid. The optimisation problem is to find the arrangement of tiles that best matches a target image. The initial population contains individuals which starts with a random positioning

Gayan Wijesinghe, Shahrul Badariah Mat Sah and Vic Ciesielski are with the School of Computer Science and Information Technology, RMIT University, GPO Box 2476V, Melbourne Victoria 3001, Australia (email: {gayan,smatsah,vc}@cs.rmit.edu.au).

¹The animations described in this paper, and a variety of others, can be found at www.cs.rmit.edu.au/~vc/evolved-images/mosaics.

of tiles as in Figure 3a. As the search proceeds the best individuals gradually show the subject as in Figures 3b-d. We discovered that when a movie is constructed using the sequence of best individuals as frames the result is very engaging and appealing animation in which the subject gradually *materialises* from random tile shuffling. If the animation is run in reverse the subject *dematerialises* back to a random tile arrangement.

The success of this approach has led us to consider what kinds of animations might result if the tiles were no longer restricted to positions in a grid but could be placed anywhere and at any orientation. This paper describes an exploration of these possibilities. In particular, our research questions are:

- 1) How can arbitrary placement of tiles in photomosaics be implemented?
- 2) How should one deal with overlapping tiles, that is, situations where a new tile is to be placed on top of one or more existing tiles?
- 3) What kinds of animations result if the tiles can be placed anywhere, but not rotated?
- 4) How do the animations change if the tiles can also be rotated?
- 5) What are the differences between grid placement and arbitrary placement in terms of how closely the target can be approximated and the computation times?

II. RELATED WORK

The idea of generating images from composition of other smaller components was originally explored in *DominoPix*, a computer graphics system attributed to Ken Knowlton [4]. Sets of dominoes were used by the system to construct pictures. Later, it was extended to using pictures as the building block through the introduction of photomosaics by Robert Silvers [5]. Silvers focuses on finding the best available matching tile images by dividing a target image into two dimensional grids and comparing it with collection of photos. He applied this technology to both artistic and commercial purposes [6]. One of his famous photomosaic was the portrait of Bill Gates that was exhibited in the National Portrait Gallery in London thus establishing photomosaic as art. Quantitative methods for generating photomosaics were first published by Tran [7]. He suggested two quantitative algorithms based on string matching techniques for generating the art. The first algorithm computes the sum of distance of all absolute differences between the red, green and blue components for each tile and selects the tile with minimum total. Meanwhile the second algorithm

applies string alignment algorithm on each row of pixels which is treated as a string and compares the score with the corresponding row in the original picture. The research had also investigated the relationship between similarity of the photomosaic with target image where it improves with larger size of library. Furthermore, it also had stated that the number of tiles used in the construction had a bigger impact on it which suggests that the diversity of tiles are more important than the size of the database. Kim and Pellacini [8] extends the idea of photomosaic to placement of arbitrary tiles shapes to fill arbitrary container image. Their technique uses an energy-based framework to calculate the sum of energy for the mosaics. Optimal tile configuration is selected based on the minimum sum. However, in order to produce good results, it requires small deformations of the tiles.

The algorithm proposed by Park [9] implements the original Photomosaic algorithm with concentration on the arrangement of the tiles where the algorithm rotated the grid and the candidate tile before stacking it to create depth in the final composition. Although the algorithm uses small size database, it seems that the algorithm can only deal with certain types of pictures orientation and this information had not been clearly stated.

Animation of mosaic tiles not only creates interesting effects to artist but also proves to be beneficial to scientific data imaging. NASA uses this technique to visualize the Amazon rainforest using tiles of satellites images [10]. Smith et al. [11] explore mosaic animations by constructing temporally coherent frames through the coordinated movement of primitive groups. They also demonstrated the use of centroidal area-based Voronoi diagram (CAVG) for tile arrangements to create the mosaic which is actually based on Hausner's work [12].

Ciesielski et al. [3] introduced the use of GAs to generate grid-based photomosaics by evolutionary searching for the best matching tiles arrangement to the target image. They animated the process of searching by using the best-fit solution from each iteration as the frame for the animation. This work focused on the grey-scaled photomosaics through two types of tiles which are generic and miniature portraits of the researchers.

III. GRID PLACEMENT

In this method, the task is to find an arrangement of tiles on a two-dimensional grid in order to maximise its closeness to a target image. As the grid is uniform and constant, only a fixed number of tiles can be placed. Therefore, this task can be formulated as an optimisation problem, suitable for an optimisation algorithm such as a GA. Our implementation uses the SGA-C [13] implementation of GA. More details can be found in [3].

A. Formulation

First, an empty canvas of the size of the target image is created and logical partitions are made on it to create the grid. The cell size is manually chosen so that it is small enough to be able to bring out the detail in the overall photomosaic but

TABLE I
GENETIC ALGORITHM CONFIGURATION FOR GRID BASED ANIMATIONS

Parameter	Value
Chromosome Length	2652 integers (Figure 1)
Chromosome Length	480 integers (Figure 3)
Population Size	200
Crossover Rate	0.7
Mutation Rate	0.0001
Elitism Rate	0.1
Generations	10,000
Target Size	1020 x 780 pixels (Figure 1)
Tile Size	15 x 20 pixels (Figure 1)
Target Size	120 x 100 pixels (Figure 3)
Tile Size	5 x 5 pixels (Figure 3)
Selection	Proportional to fitness
Replacement	Generational replacement

large enough to clearly show the details of each tile when viewed up close. Additionally, the cell size must lead to an integer number of cells along with the width and the height of the image. The numbers of rows and columns of the grid are determined by dividing N , the width of target image and canvas by n , the width of a tile and M , the height of the target image by m , the height of a tile. $N/n = R$ tiles are required to fill a row and $M/m = C$ tiles to fill a column. A potential solution is an $R \times C$ array of integers representing an index from t tiles. The GA uses this matrix represented as a vector in row order as the chromosome.

B. Fitness Evaluation

The fitness of a chromosome is evaluated by calculating the sum of the pixel differences between the candidate photomosaic and the target image (equation 1). The global best fitness in this method remains unknown, leaving some error, since a photomosaic can never be perfect copy of the target at the pixel level.

$$\sum_{i=1}^N \sum_{j=1}^M |target(i, j) - individual(i, j)| \quad (1)$$

Due to the nature of the grid, pre-computation of certain pixel calculations can be done to speed up the fitness evaluation process. A table is constructed of the error that occurs when each tile is placed in each cell of the grid. Fitness evaluation then becomes mostly a matter of table lookup rather than summation of pixel differences. Table I shows the basic configuration used for Figures 1 and 3. The parameter values were determined through preliminary experimentation.

IV. ARBITRARY PLACEMENT

Arbitrary placement of tiles introduces considerably more complexity into the task. The number of tiles to be placed is no longer a constant but will vary between individuals. It is now possible for a tile to be placed in an area of the canvas where one or more tiles have already been placed. A background colour needs to be chosen for the areas of the canvas where no tile has been placed. The fixed length

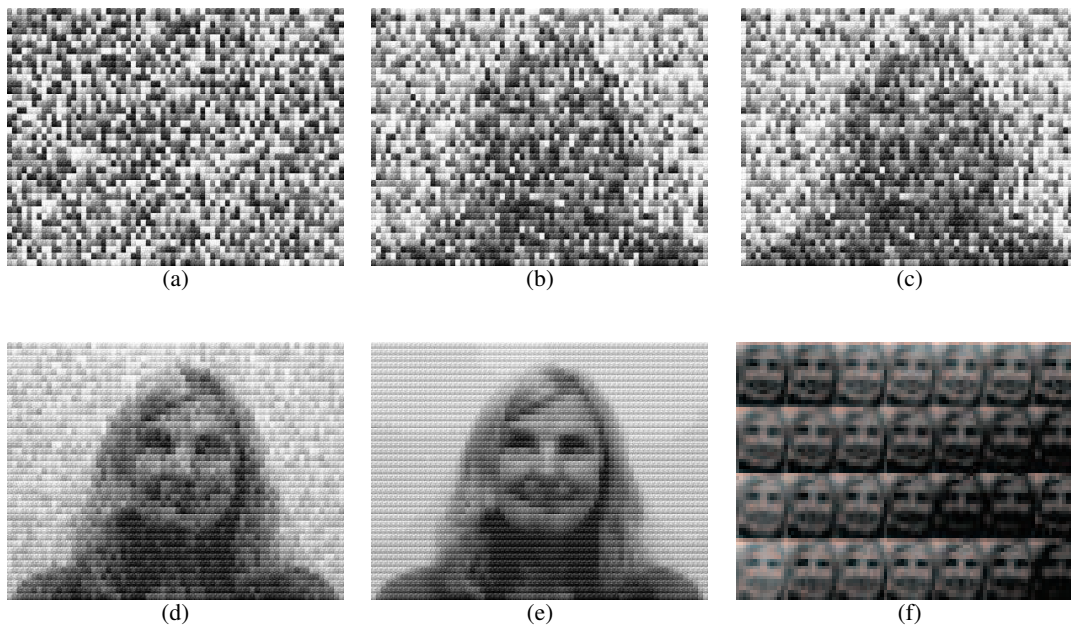


Fig. 1. Examples of an Evolved Sequence. Image (f) is a small segment of image (e) shown at actual size.

chromosome representation used for the grid approach is no longer possible.

In our formulation of the arbitrarily placed tiles problem, the search must find the tiles to use as well as their placement on the canvas. To facilitate implementation and experimentation we have implemented arbitrary placement using tree based genetic programming. The configuration is summarized in Table II. There is a terminal for placing a tile at a particular position on the canvas and a join function (PROG3 in the terminology of [14]) for sequentially executing 3 commands. These commands can be tile placement commands or recursive join commands. The tile placement terminal randomly generates a tile number t and places it at a randomly generated position (x, y) . Each individual thus encodes a variable number of tiles to be placed on the canvas. The maximum number of tiles is determined by the tree depth. We have used the RMITGP package for the implementation [15].

The fitness evaluation, again, is the pixel-level closeness of the canvas to the target image, as done in the grid-based approach. An interesting preliminary observation was made on the placement of tiles on the canvas. Particularly, the grid-based method requires some tiles to be placed at every cell on the grid but the arbitrary placement method places tiles only where they are required, leaving the potential for empty space on the canvas. How one deals with this can make a significant difference in the animation. We discovered that a search on a canvas initialised with the average colour value of the target leads to a high level of fitness relatively early compared with searches that started with a black or white coloured canvas. However, this caused an early convergence

of the search, producing uninteresting results. Therefore, a black canvas was used for the subsequent experiments.

In order to investigate the effectiveness of the search in this arbitrarily placed problem, we first investigate a simplified variation which does not have tile rotations and the tiles are small uniform blocks. After this we look at the same target and tiles as used in the grid animations of Figure 1.

A. Non-Rotated Tiles

As this is the first attempt at the arbitrarily placed tile problem, flat-shaded tiles are used to investigate the effectiveness of the generated photomosaics. A terminal representing tiles and a function that allows three tiles to be linked together are used in this investigation. The terminal contains information of tile ID and a random coordinate on the canvas. Tile size of 5 by 5 pixels and target size of 120 by 100 pixels have been employed in this investigation. We tested three different tile placement rules, which resulted in significantly different outcomes. Their details are explained in the sections below. The GP configuration used for this approach is shown in Table II.

1) *Rule 1: Fitted Placement:* When using this rule, a tile is placed on the canvas only if it is not going to cover any other existing tile on the canvas. It requires an empty space on the canvas that is as large as the tile to be placed. The first frame, some intermediate frames and the final frames of the evolutionary process, when this rule was used, are shown in Figures 4a-d. The images have been selected to display the relationship between a fitness value and the visual appearance of the evolved individual. As expected, this rule leads to many empty spaces on the canvas and causes many tiles in

TABLE II
GP CONFIGURATION FOR NON-ROTATED TILE PLACEMENT

Parameter	Value
Population Size	200
Max Generations	20,000
Crossover Rate	0.70
Mutation Rate	0.25
Elitism Rate	0.05
Max depth	9
Min depth	2
Terminal	Place tile t at (x, y)
Function	join(Place tile or join, Place tile or join, Place tile or join.)

the evolved individuals to be not placed on the canvas. Using this rule it was not possible to achieve high fitness scores. In fact, rarely was the subject evident in the final best individual. In theory it should be possible to achieve the same kinds of animations as the grid approach, but the probability of this happening is very small.

2) *Rule 2: Conditional Placement:* When this rule is applied, a tile is only placed on the canvas if the area to be covered by it is less than half full of pixels of other tiles. This adds a level of control of the amount of overlapping allowed. For simplicity, the overlapping area is overwritten by the latest tile placed on the canvas. Figures 5a-d show frames extracted from the start, middle and the end of the evolutionary process of a run where this rule was applied. With some analysis we found that a large number of tiles of most solutions did not get placed on the canvas due to the conditional placing, making this rule's solutions bloated, similar to the solutions obtained with Rule 1. However, it was possible to achieve better fitness than with rule 1, with just a vague suggestion of the target in the best individual.

3) *Rule 3: Conditional Random Placement:* With this rule, a tile is only placed if the area it is meant to cover on the canvas is empty. If there are any parts of other tiles in the selected position, the new tile is placed at a randomly chosen coordinate. Here, we added the extra random element in the tile placement to see its effect on the animated subject formation. Figures 6a-d show the first, several intermediate and final frames extracted from the animation of this evolutionary process. When compared to the results of Rule 2, we found that the fitness differences of the final solutions were significant. Figure 2 shows the best fitness values over 20,000 generation for each rule. The results show that changes are more frequent in early generations and it also reveals rule 3 as the best rule based on the best fitness values. The animation of this photomosaics create an interesting visual effects, however the composition might not be repeatable as many random elements are introduced to the program.

B. Rotated and Blended Tiles

In this experiment, we extend our previous method of placing tiles at arbitrary locations by adding additional parameters to optimise. Specifically, tiles now can be rotated up

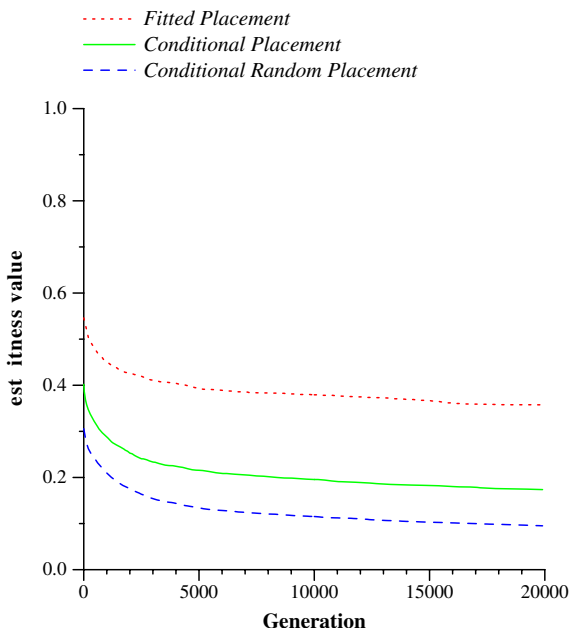


Fig. 2. Best Fitness values for non rotated tile

to 360 degrees about their centers and they are added on to the canvas with a level of transparency ranging from 20% up to 80%, allowing blended, much more complex photomosaics to be formed. Additionally, we use 20x20 pixel textured tiles instead of the flat-shaded tiles used in the previous section to further the visual complexity of the generated subjects. Each tile contains a picture of a person's face, as does the target image. The target image is relatively large at 1020x780 pixels in size, same target that was used for Figure 1. This setting was selected based on the settings of previous grid-based placement study that produced photomosaics as in Figure 1. A total of 5 tiles were used, however with the transparency values, they provide a much greater level of variations to form subjects. Figures 7a-d show the first, intermediate, and final frames from the evolutionary run. The final frame was obtained after 18,000 generations. Due to the much richer set of allowed operations, the final photomosaic has a greater depth and the animated evolutionary process highly engaging. However, the evolution required a modern PC to run several days to run as opposed to several hours needed for the other place-anywhere methods.

V. CONSTRUCTION OF ANIMATIONS

To build the animation, the best solution at each generation is rendered to consecutive frames in the animation. To achieve the *materialising* and *dematerialising* effect described earlier, the first half of the animation has the frames arranged in the ascending order of generations and the second half is the reverse of the first half.

We found that different viewers may render the same animation file differently. Very pleasing artistic effects achieved

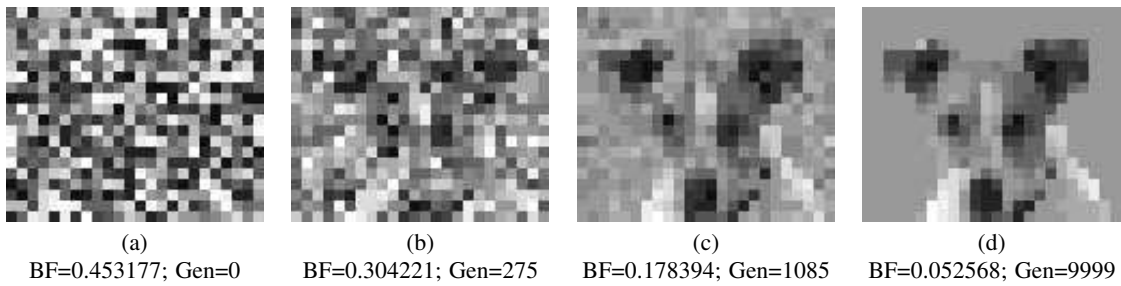


Fig. 3. Sample frames from the grid-based approach. Note the best possible fitness is 0.0

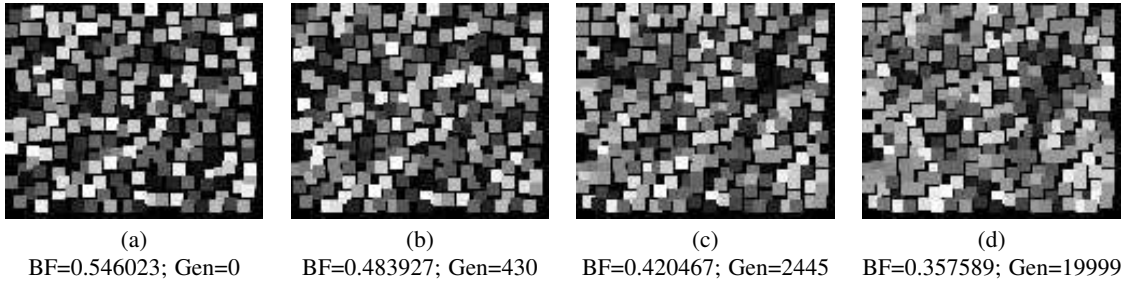


Fig. 4. Sample frames from the arbitrarily-placed, non-rotated, fitted tile placement approach with best fitness values and generation numbers

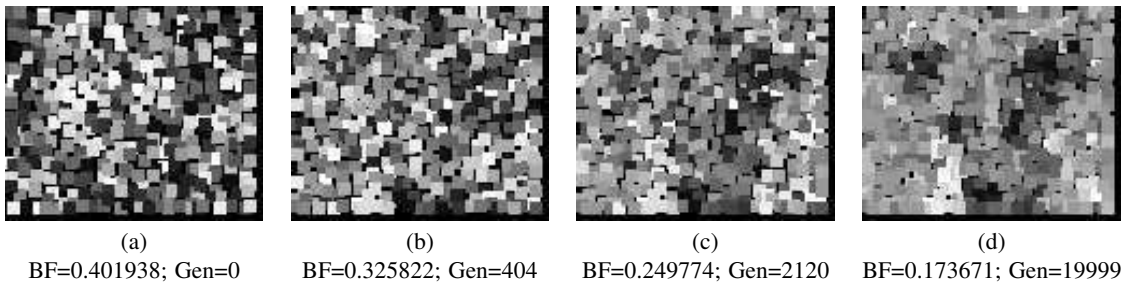


Fig. 5. Sample frames from the arbitrarily-placed, non-rotated, conditional tile placement approach with best fitness values and generation numbers

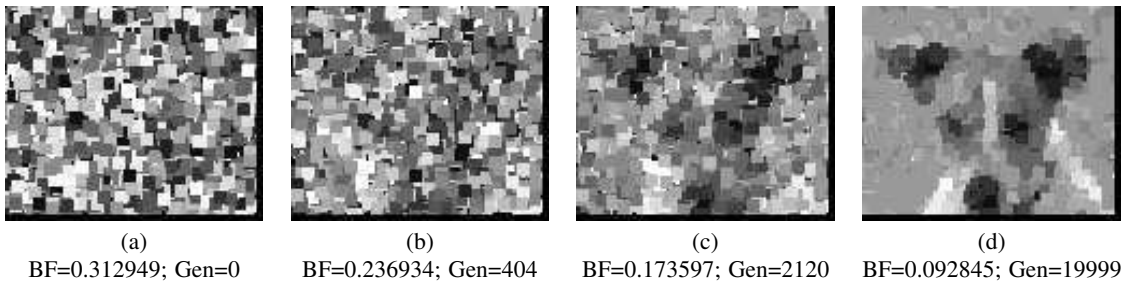


Fig. 6. Sample frames from the arbitrarily-placed, non-rotated, conditional random tile placement approach with best fitness values and generation numbers

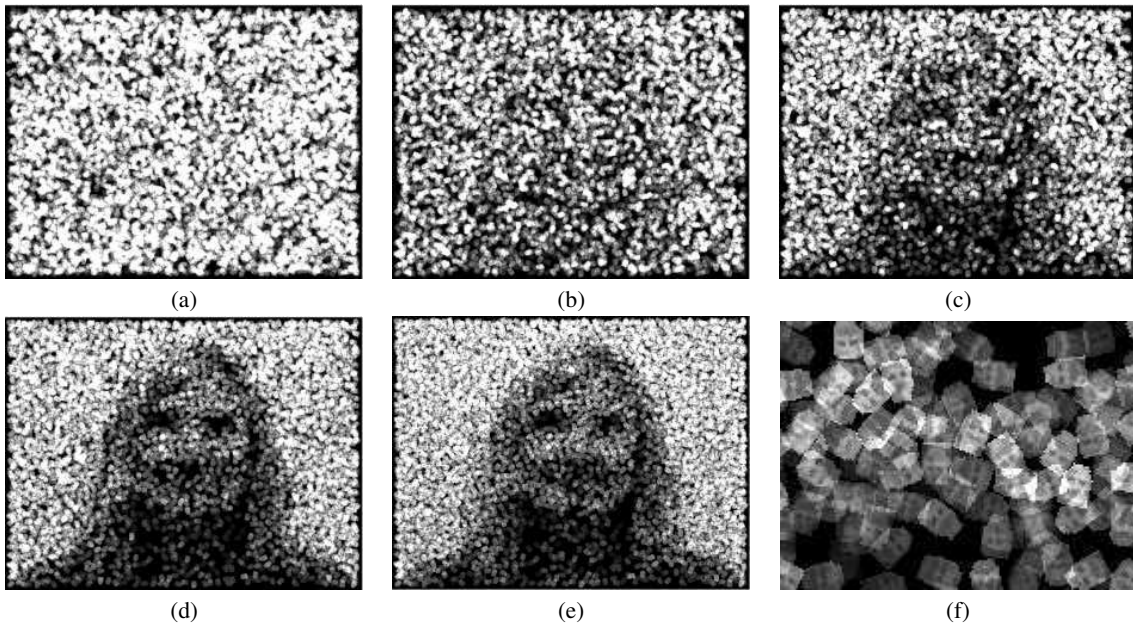


Fig. 7. Sample frames from the arbitrarily-placed, rotated and blended tile placement approach. Image (f) is a small segment of image (e) shown at actual size.

with one viewer were lost with another viewer. The speed of the computer processor could also affect the rendering. For this experiment, we constructed the MPEG files using PPMTO MPEG. Viewing the movies with the VLC player under Microsoft Windows, the MPLAYER player on Solaris or the Totem player in the Linux environment retain the artistic visual effects.

VI. COMPARISON

A. Artistic Style

While it is difficult to describe artistic style, the grid based and the place-anywhere animations are qualitatively different. The grid based animations reach a closer likeness to the target, but the place-anywhere animations have a richer, more complex feel to them. The grid-based animated photomosaics have a distinct shuffling effect compared to the place-anywhere ones. As the tiles in the arbitrary arrangement allow overlap, the effects of switching the tiles appear to be less visible. However, both animations are unique and very engaging as the subject of the photomosaics start to emerge and then returns to random. The best images of the grid based approach contain many uniform areas of repeated tiles, while this is totally absent in the place-anywhere approach. Our current place-anywhere animations contain areas that are saturated to white. This suggests we should re-examine our transparency parameters. However, against a black background the effect is quite striking.

B. Computational Effort

The computational effort for evaluating an individual in the place-anywhere approach is significantly higher. In the grid

approach, it is possible to precompute components of the fitness in a pre-processing step, as described in section III-B, thus saving considerable time in evaluating an individual. In the place-anywhere approach with transparency, this is not possible. Furthermore, the grid approach need take no notice of what is already on the canvas while additional computation is required in the place-anywhere approach to determine whether any pixels of the target area are already occupied and what to do about it.

C. Size of Search Space

For the grid based approach the size of the search space is given by:

$$(R \times C)^t \quad (2)$$

where R is the number of rows of tiles, C is the number of columns of pixels and t is the number of different tiles.

For the place-anywhere approach, ignoring for simplicity the tile blending, the number of possibilities is

$$(w \times h)^{360t} \quad (3)$$

for a $w \times h$ pixel image with t tiles. Clearly $w \geq R$ and $h \geq C$ so there are many orders of magnitude more possibilities for the place-anywhere approach. This situation is exacerbated by our representation which has a many-to-one genotype-phenotype mapping.

VII. CONCLUSIONS

Our main goal in this work was to determine whether aesthetically pleasing animations could be evolved using a place-anywhere strategy rather than a grid-placement strategy

for the tiles. In this we have succeeded. We have generated a number of visually engaging animations using a place-anywhere strategy. With respect to our specific research questions we have the following outcomes: 1. *How can arbitrary placement of tiles be implemented?* This can be done using a genetic programming formulation. This approach facilitates experimentation and variation, but this comes at the expense of additional computational requirements. 2. *How should one deal with overlapping tiles, that is, situations where a new tile is placed on top of one or more existing tiles?* The best option we found was to blend newly placed tiles with those already on the canvas. However there are many possibilities here and more exploration of placement strategies and the consequent artistic effects is needed. 3. *What kinds of animations result if the tiles can be placed anywhere, but not rotated?* Such animations are more interesting than the corresponding grid based animations but come at the expense of considerably more computing time. 4. *How do the animations change if the tiles can also be rotated?* The animations have a richer, more interesting texture, but at more computational cost. 5. *What are the differences between grid placement and arbitrary placement in terms of how closely the target can be approximated and the computation times?* The grid placement approach results in closer matches to the target in many fewer generations. Also, the cost of evaluating an individual is much lower because pre-computation of the error resulting from placing each tile in each grid cell is possible.

This work has suggested further avenues of exploration. A non-redundant representation, more suited to this specific problem, is needed. More work is needed on strategies for placing tiles on the canvas and blending with any tiles already placed. It would also be very interesting to see whether having a variable tile size would lead to qualitatively different animations.

Acknowledgements

We thank our colleagues in the RMIT Evolved Art Group, particularly Marsha Berry, Karen Trist and Daryl D'Souza, for their helpful suggestions and ideas relating to this work.

REFERENCES

- [1] Anon. The War President. <http://www.matrixmasters.com/world/usnews/WarPresidentMosaic.html>, Visited 19-Feb-08.
- [2] Vic Ciesielski, Marsha Berry, Karen Trist, and Daryl D'Souza. Computer Animation: Constantly Becoming Other. In *Art Escapes: Variations of Life in the Media Arts, Exhibition Catalog*, pages 13,52–55,113. Universitat Politecnica de Valencia, Spain, Apr 2007.
- [3] Vic Ciesielski, Marsha Berry, Karen Trist, and Daryl D'Souza. Evolution of animated photomosaics. In Marco Giacobini, editor, *Applications of Evolutionary Computing: Proceedings of the 5th European Workshop on Evolutionary Music and Art (EVOMUSART07)*, volume 4448 of *LNCIS*, pages 498–507. Springer, April 2007.
- [4] Ken Knowlton. DominoPix, US Patent No. 4,398,890, Representation of Designs. <http://www.knowltonmosaics.com/>, <http://www.metron.com/DominoPix/>, 1983. Visited 02-Nov-06.
- [5] Robert Silvers and Micheal Hawley. *Photomosaic*. Henry Holt and Company, Inc., New York, 1997.
- [6] Runaway Technology. Company web site. <http://www.photomosaic.com/rt/highlights.htm>, Visited 06-Oct-06.
- [7] Nicholas Tran. Generating photomosaic: An empirical study. In *Proceedings of ACM Symposium on Applied Computing SAC' 99*, pages 105–109. The Association for Computing Machinery, February 1999.
- [8] Junhwan Kim and Fabio Pellacini. Jigsaw image mosaics. *ACM Transactions on Graphics (TOG)*, 21(3):657–664, 2006.
- [9] Jin Wan Park. Artistic depiction: Mosaic for stackable objects. In *ACM SIGGRAPH 2004 Sketches SIGGRAPH '04*. The Association for Computing Machinery, August 2004.
- [10] NASA. Mapping the Amazon: Mosaic tiles animation. <http://svs.gsfc.nasa.gov/vis/a000000/a002400/a002405/>, Visited 06-Oct-06.
- [11] Kaleigh Smith, Yunjun Liu, and Allison Klein. Animosaic. In *Proceeding of Eurographic/ACM SIGGRAPH Symposium on Computer Graphics 2005*, pages 105–109. The Association for Computing Machinery, July 2005.
- [12] Alejo Hausner. Simulating decorative mosaics. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques SIGGRAPH '01*, pages 573–580. The Association for Computing Machinery, August 2001.
- [13] Robert E. Smith, David E. Goldberg, and Jeff A. Earickson. SGA-C: A C-language implementation of a simple genetic algorithm, September 05 1991. <http://citeseer.ist.psu.edu/341381.html>.
- [14] J. R. Koza. *Genetic Programming: On the Programming of Computers by the Means of Natural Selection*. MIT press, Boston, 1992.
- [15] Vic Ciesielski. The rmit-gp genetic programming system. Available from <http://www.cs.rmit.edu.au/~vc>, 2006.