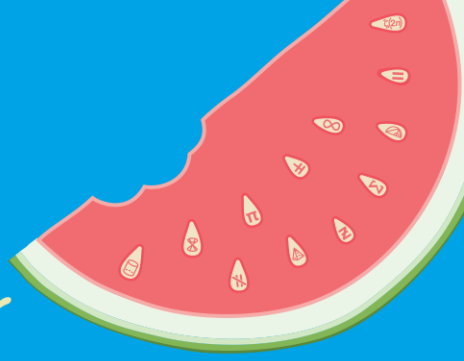
AMSI VACATIONRESEARCH SCHOLARSHIPS 2021–22

Get a taste for Research this Summer



Optimising the Image Mosaicing of Visual Representations of Functions

Hannah Perry

Supervised by Prof Kate Smith-Miles and Dr Andrés Muñoz

The University of Melbourne

AMSI AUSTRALIAN MATHEMATICAL SCIENCES INSTITUTE

Contents

Abstract

Building on previous work exploring computational construction of mathematical artworks, we developed alternative algorithms for arranging image tiles into non-overlapping mosaics aiming to enhance or destroy 'blue river connectivity' in the final mosaic. We explored applying simulated annealing to the greedy swap algorithm and an algorithm based on generating and joining graphs. Simulated annealing did not have a significant effect in comparison to the greedy swap algorithm alone. The graph generating algorithm appeared reasonably successful in connecting rivers and forming larger groupings of blue, however, was not as successful in generating discrete linear forms of rivers as was seen in manual swapping.

Introduction

This report focuses on algorithms to automate the construction of non-overlapping mosaics from image tiles of contour plots of functions. In this context, mosaicing refers to the placing of image tiles in an arrangement to create a larger image.

Recent work by Smith-Miles and Muñoz [1] explored the computational construction of mathematical artworks from the arrangement of visual representations of functions. The functions were composed as part of Smith-Miles' ARC Laureate project MATILDA [2] by using a genetic programming algorithm on a set of operations, to generate a more comprehensive set of functions that span across the instance space for continuous black-box optimisation problems [3]. This results in the functions having distinctive appearance in visualisations and complicated minima and maxima. The visual representations in [1] are given a blue to yellow colour palette, which gives the blue minima the appearance of rivers when they connect in the mosaic of these images. In [1], the authors created an artwork based on manually forming and destroying blue river structure and asked whether an algorithm can achieve the same goal but faster. This requires a mathematical definition of 'blue river connectivity' to ask an algorithm to enhance or destroy. The results presented in [1] showed only limited success in achieving the artistic goal using an algorithm compared to the human effort.

This project extends on the work completed by Smith-Miles and Muñoz by refining aspects of the algorithm used, as well as developing a new algorithm and investigating alternatives for optimising the river connectivity. We present the results from applying simulated annealing to the greedy algorithm, as well as the development of a cell-based tile simplification and graph-theoretic approach to form river connectivity in the mosaics.

In Section 1, we present the previous work, in Section 2 we introduce the results of our simulated annealing, then in Section 3 we present our graph based algorithm. Finally, we present our conclusions and discussion.

Statement of Authorship

Smith-Miles conceptualised the project. Smith-Miles and Muñoz produced the function visualisations. Muñoz wrote the original software (MATLAB). Perry implemented simulated annealing in the existing software, and wrote the graph generating algorithm software in Python. Perry wrote this report, which was reviewed by

Smith-Miles and Muñoz.

1 Previous Research

1.1 Background

The following is a brief summary of the previous work that forms the foundations of this report. Previous work by Smith-Miles and Muñoz, explored the optimisation of non-overlapping mosaics using individual image tiles that are visual representations of functions. These visual representations were created by taking contour plots of functions. These functions are generated via a genetic programming algorithm, initially intended for stress-testing optimisation algorithms in [3].

A colour mapping was applied to the contours to define them in a 2-dimensional space. The blue to yellow colour map (Parula from MATLAB) gives the minima the appearance of forming rivers.

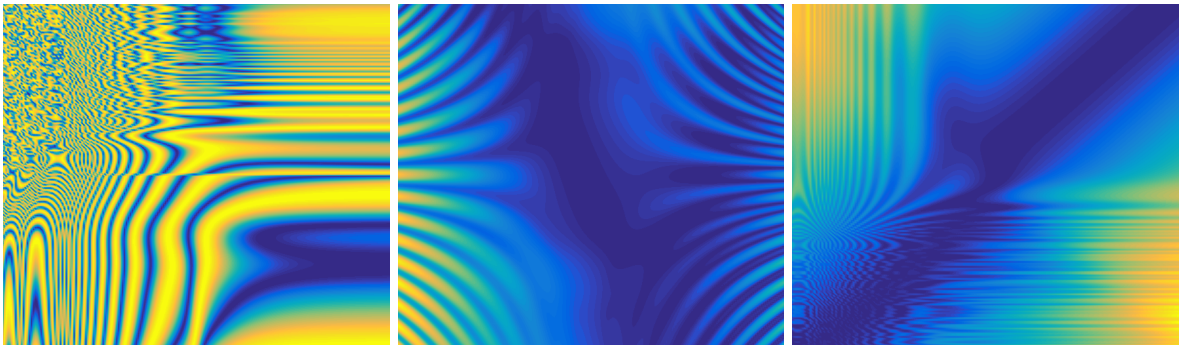


Figure 1: Examples of the contour plots of a selection of functions from Muñoz and Smith-Miles [3]

A mosaic can be constructed by placing the 306 image tiles into an 18 by 17 grid. In figure 2, which is an example of such a mosaic, by chance the random arrangement contains some regions where the deep blue continues across adjacent tiles, forming accidental 'river' structures. These 'river' structures draw the eye to particular sections of the mosaic, which may be aesthetically desirable or undesirable depending on individual preference.

1.2 Artwork

In [1] the authors create an artwork, Negentropy Triptych (figure 3), that arranged the tilings of 306 images into a triptych of 3 mosaics of 18×17 images. The centre mosaic is a random arrangement, while the left mosaic used around 30 manual swaps to destroy the accidental background blue river connectivity, and the right mosaic used around 30 swaps to enhance the blue river connectivity. The result is an artwork that represents the spectrum of aesthetic taste from disordered to ordered. Given this objective, the authors asked whether an algorithm can achieve the same goal faster, but this requires a mathematical definition of 'blue river connectivity' to ask an algorithm to enhance or destroy.

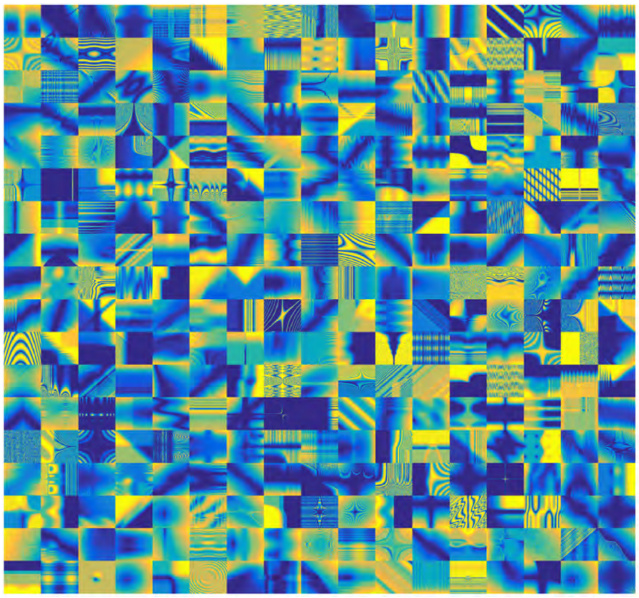


Figure 2: Original random arrangement of the 306 contour plot images, source: Smith-Miles and Muñoz [1]

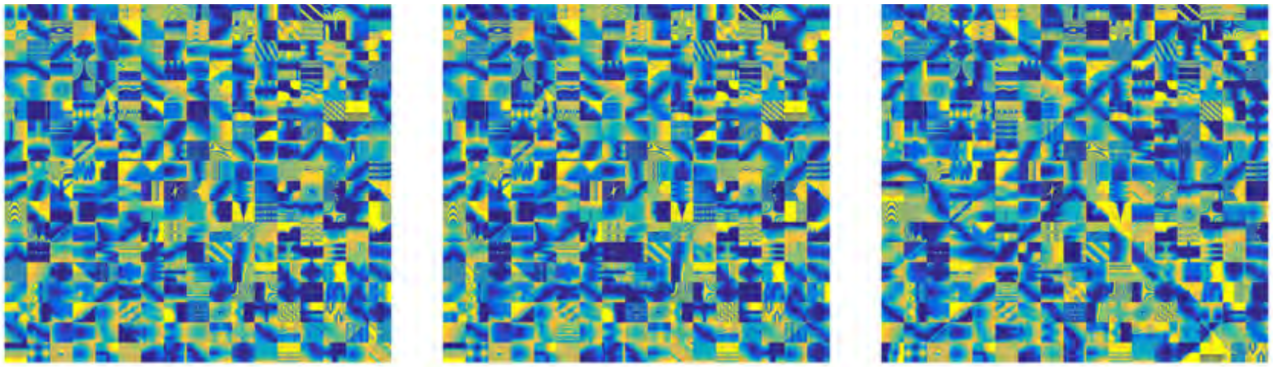
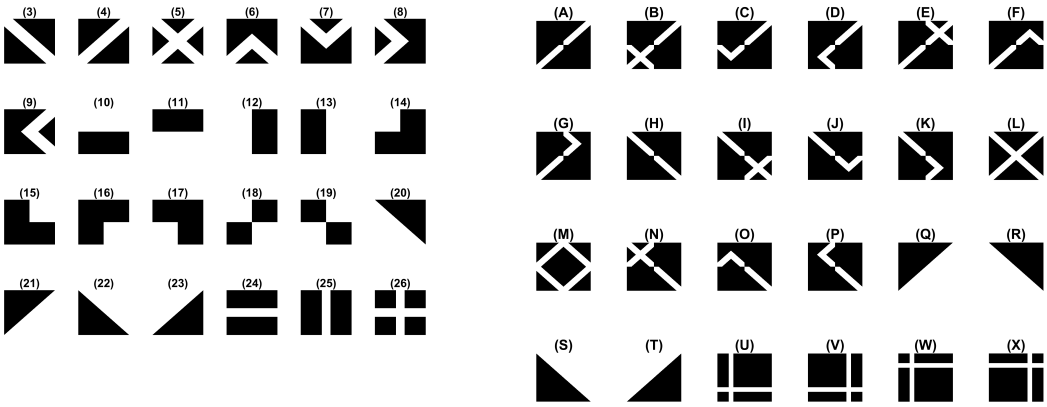


Figure 3: 'Negentropy Triptych', digital, Kate Smith-Miles and Mario Andrés Muñoz, copyright 2019, [1]

1.3 Greedy Swap Algorithm

Previously in [1] a greedy algorithm that used seven neighbourhood swap operators was used to automate the destruction and creation of the river structure, intended to emulate the process of manually swapping images. The swap operators included swapping two randomly chosen images, swapping a randomly selected image with its neighbour in the direction of left, right, upper or lower, and replacing a randomly selected image with its horizontal or vertical mirror image. At each iteration the algorithm would randomly select an operation to apply to create a new mosaic, calculate the value of the objective function of the new mosaic, and compare to the previous iteration to base the decision of whether to accept or reject the new mosaic. Various objective functions were tried, based on measures of mutual information (pixel location and the change on intensity), connected pixels (maximising the largest area of connected pixels in the binary image using `BWAREAFILT` MATLAB function), and pattern identification (measure of similarity between window of primitive shapes, figure 4).



(a) Primitive shapes that represent the strongest deep blue feature from an individual image

(b) 2 by 2 Pattern windows

Figure 4: Pattern identification

1.4 Results

The resulting images (figure 5) are not fully successful in showing the kind of blue river connectivity that can be achieved by manually swaps that this project is trying to replicate. One possible cause of the algorithm not succeeding in that aim could be its greediness leading to being restricted to finding a local minima or maxima. As suggested in [1], more sophisticated techniques such as simulated annealing could be used to find a better optimisation.

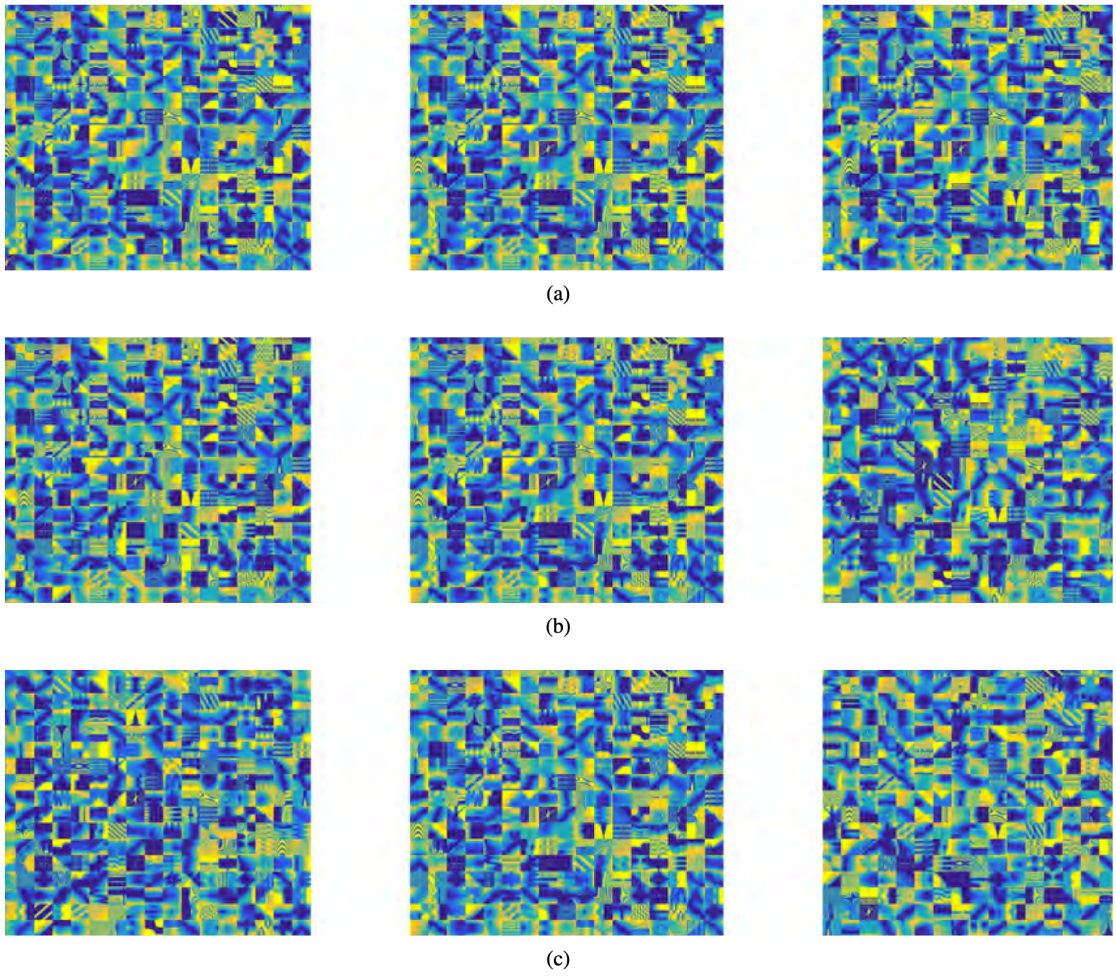


Figure 5: The mosaics generated by the swap algorithm, left column minimised, centre preserved, right maximised, for (a) mutual information, (b) connected pixels and (c) pattern identification objective functions, source: Smith-Miles and Muñoz [1]

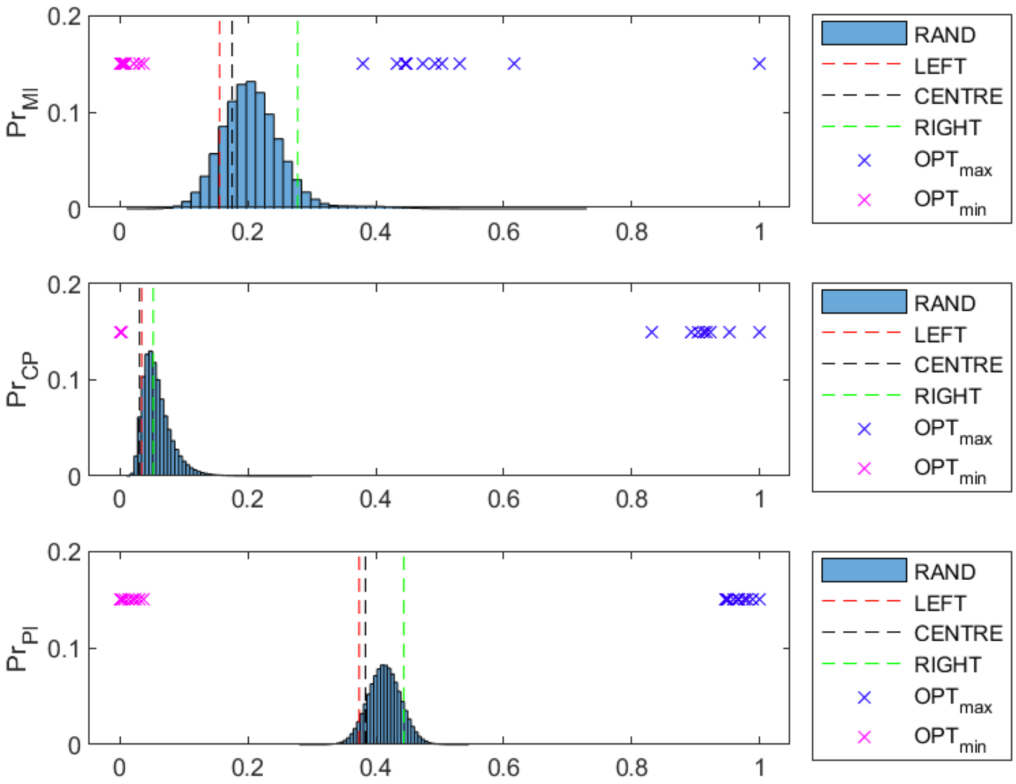


Figure 6: Results from the greedy swap algorithm: 10 maximisation runs, 10 minimisation runs, compared with 10^6 randomly generated, for each of the 3 objective functions. Source: Smith-Miles and Muñoz [1]

2 Simulated Annealing

2.1 Motivation

To explore the effect of using a non-greedy algorithm, we implemented simulated annealing into the existing code from [1]. Simulated annealing is a technique that aims to improve hill climbing ability by temporarily accepting a worse solution than the current solution. This can be useful to avoid getting stuck in local minima or maxima if the objective function is non-convex.

2.2 Method

The probability of accepting a worse solution depends on objective function at current solution e, solution being considered e_{new}, and current temperature T.
For the case when the objective function is to be maximised, the probability of accepting an iteration is given as,

$$P(e, e_{new}, T) = \begin{cases} 1 & e_{new} > e \\ \exp\left(\frac{e_{new}-e}{T}\right) & e_{new} \leq e \end{cases}$$

For minimisation,

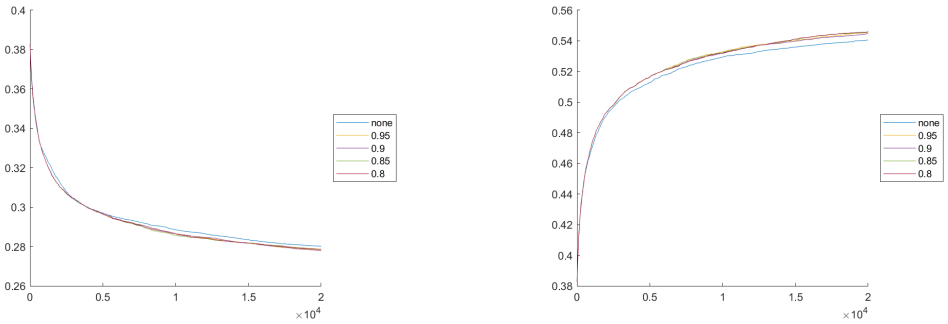
$$P(e, e_{new}, T) = \begin{cases} 1 & e_{new} < e \\ \exp\left(\frac{-(e_{new}-e)}{T}\right) & e_{new} \geq e \end{cases}$$

We tested different initial temperatures and cooling schedules in an attempt to explore the possible impacts on the effectiveness of simulated annealing for this application.

2.3 Results

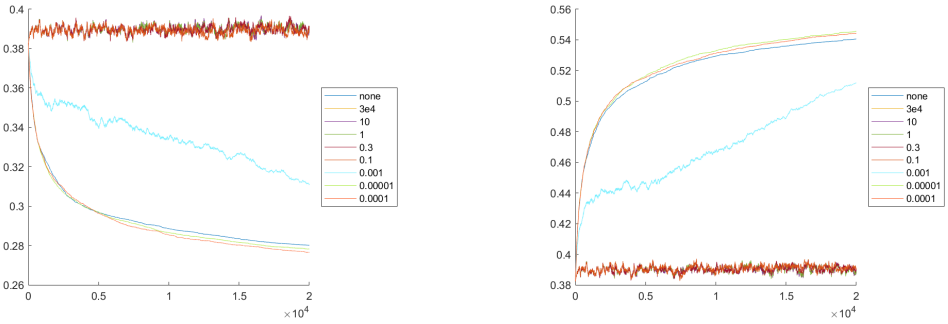
Simulated annealing did not have a significant impact on the value of the objective function at the solution found. Visually the river connectivity in the final mosaics by this method were very similar in appearance to the mosaics created by the greedy algorithm for both creation and destruction.

There are several possible explanations for why the approach of simulated annealing did not improve the solutions. The greedy approach may have been finding a very decent optimal, given the objective functions. It is also possible that the objective functions previously trialled may not be capturing the river connectivity concept sufficiently to allow for meaningful visual improvement.



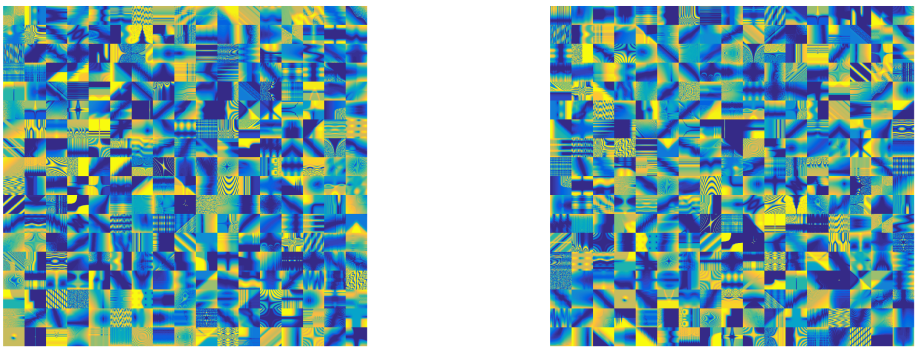
(a) Minimisation (b) Maximisation

Figure 7: Plot of the average objective function of 10 runs across 20000 iterations of the algorithm, comparing different alpha values (a parameter that controls the speed of cooling)



(a) Minimisation (b) Maximisation

Figure 8: Plot of the average objective function of 10 runs across 20000 iterations of the algorithm, comparing simulated annealing with various starting temperatures to no simulated annealing



(a) Minimisation (b) Maximisation

Figure 9: An example of final mosaic after 20000 iterations with simulated annealing, initial temperature of 0.00001, alpha of 0.8

3 Graph Generating Algorithm

3.1 Motivation

We considered a new approach, of an algorithm that builds up the mosaic, and an objective function that uses graph theory to abstract the blue density of sections of each image tile and looking at paths through adjacent images. While developing this, we aimed to create a computationally efficient heuristic algorithm that considers all images currently placed in the mosaic.

3.2 Algorithm

3.2.1 Comparison of Methods to Identify Blue Areas

Preprocessing of each individual image tile to develop a representation that is compatible with being turned into a graph is an essential first step in this new approach. This involved identifying where the deep blue values occur in each image tile, for which the hue value gave the best indication of blue-ness. In order to reduce computation time of the later stages of the algorithm, it was essential to downscale the complexity of each individual image tile.

Previously, in the pattern identification method of the greedy swap algorithm, the simplification of image tiles to their primitive images was quite successful at reducing the complexity of the images and identifying where the rivers occurred in the image tile.

In endeavouring to improve on the detection of continuous rivers across the image tile by assessing hue values, we compared two preprocessing methods as shown in the examples in figure 10.

Method 1 involves resizing the images using a general image processing library function that utilises average pixel value, with interpolation by area. Then considering whether each pixel is within the blue hue range.

Method 2 divides the image tile into ‘cells’, and looks at the proportion of pixels in each cell that are within the deep blue hue range. If the proportion of deep blue pixels is sufficient then we characterise that the cell contains ‘water’.

For each of these methods, a 5 by 5 resolution per image appeared to have reasonable complexity, while not losing too much definition of the image, but without overburdening computation time. This approach might allow for the potential to be generalised to other image sets.

Ultimately, method 2 gave the desired balance of accuracy and detection of the strongest blue hue.

3.2.2 Graph Construction

Having preprocessed the tiles using method 2, a graph from each image can be obtained by considering the cells that contain enough ‘water’ (deep blue) to be nodes, which share an edge when adjacent (by 8-connected definition), as shown in figure 11. Then the shortest paths can be calculated between each cell node within the image. We are interested in finding the maximum of the shortest paths from a particular cell node.

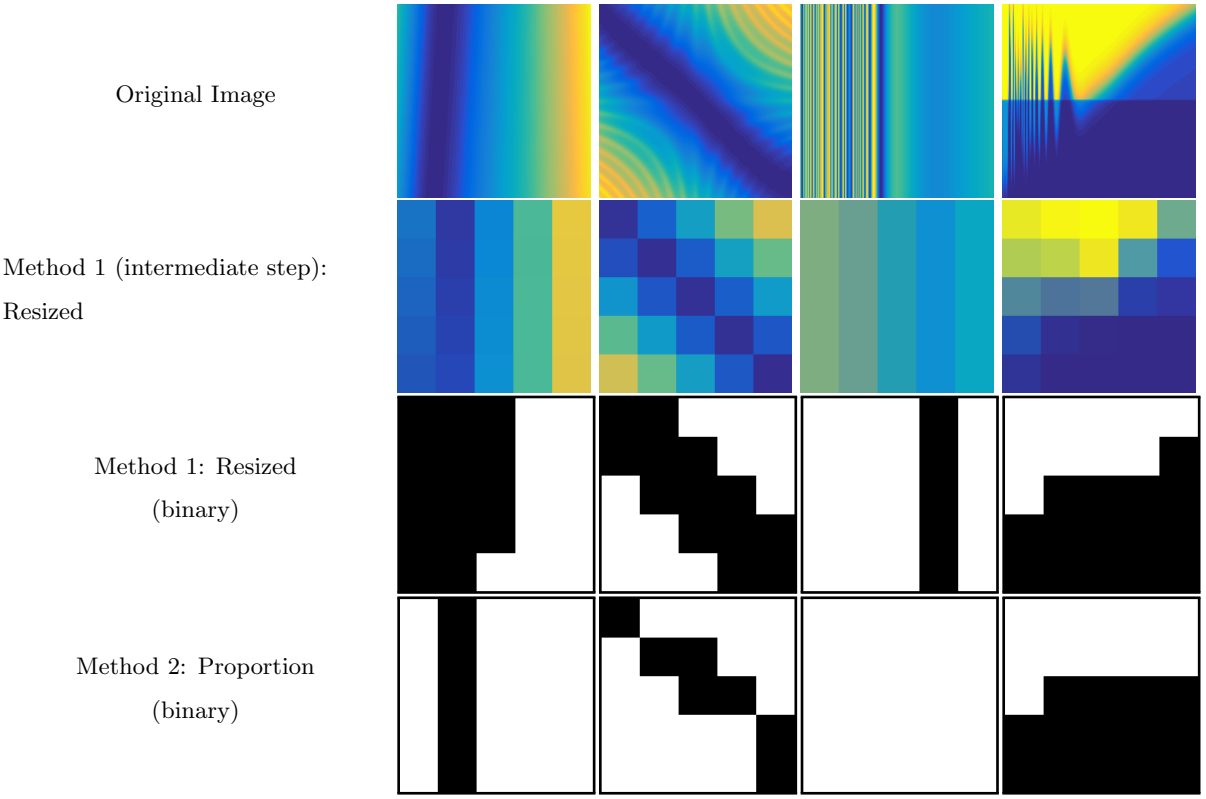


Figure 10: Preprocessing examples

Using the information derived from the maximum shortest paths transforms the task from placing images into the mosaic to deciding which graphs should be joined together to create the collective graph that represents the mosaic. By using the maximum shortest path method it attempts to encourage the formation of rivers in adjacent tiles.

The objective function involves finding the maximum shortest connected path through the cells in each pair of adjacent image tiles, then summing this across all adjacent pairs of image tiles in the mosaic.

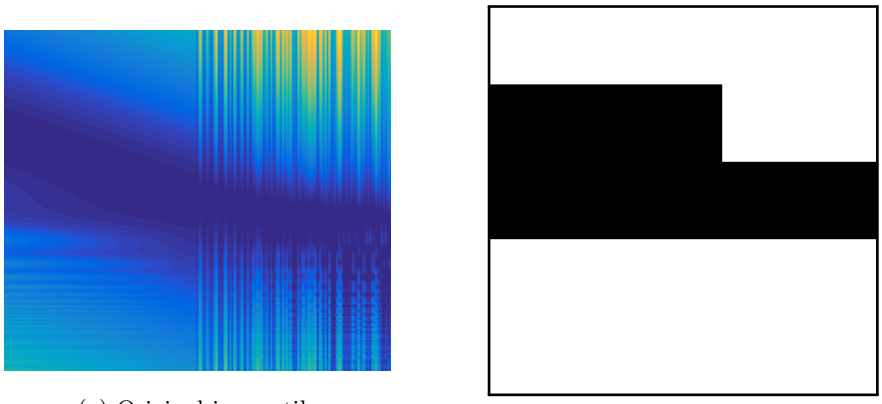
$$J_{GG}(\mathbb{X}) = \sum_{i=1}^{n}\sum_{j=1}^{m} S_{i,j}$$

Where $\mathbb{X}$ is the entire mosaic, n and m are the number of images in each row and column of the mosaic. And $S_{i,j}$ is calculated as,

$$S_{i,j} = \sum_{\mathbb{X}_{k,l} \in \text{ image tiles adjacent to } \mathbb{X}_{i,j}} C(\mathbb{X}_{i,j}, \mathbb{X}_{k,l})$$

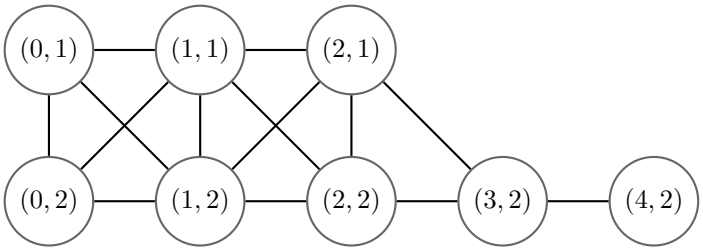
$$C(\mathbb{X}_{i,j}, \mathbb{X}_{k,l}) = \begin{cases} \max\{PL(v) + PL(u) | v \in A_{\mathbb{X}_{i,j}}\, u \in A_{\mathbb{X}_{k,l}}, v, u \text{ adjacent}\} & \mathbb{X}_{i,j}, \mathbb{X}_{k,l} \text{ non empty} \\ 0 & \mathbb{X}_{i,j} \text{ or } \mathbb{X}_{k,l} \text{ are empty} \end{cases}$$

Where $A_{\mathbb{X}_{i,j}}$ is the set of cells along the relevant border of image $\mathbb{X}_{i,j}$. $PL(v)$ is the maximum shortest path length of node v in the graph obtained from the cells.



(a) Original image tile

(b) Cells



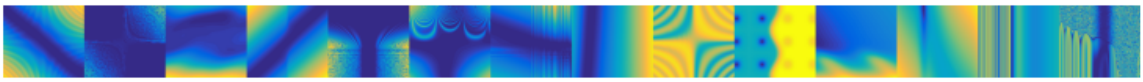
(c) Graph obtained from the cells

Figure 11: Example of construction of the graph for this image tile. For this example the maximums of the shortest path lengths from each border cell are given by: `(0, 1): 4, (0, 2): 4, (4, 2): 4`

Calculating the objective function based on adjacent image tile is more efficient than calculating global path lengths and there is potential for local updates to the objective function rather than recalculating it for the entire mosaic when a single tile is considered.

The example in figure 12 shows a subset of the image tiles being arranged in a single line from left to right based on a modified version of the objective function.

(a) Available images before arrangement



(b) Arrangement after applying algorithm

(c) Where the algorithm sees the rivers

Figure 12: Applying the objective function to select the next best image to be placed to the right

3.2.3 Choosing the Next Location to Expand

The algorithm begins with a random image in the centre of the mosaic, and grows the mosaic greedily by choosing a location to insert an image tile by picking the empty location adjacent to a non-empty location that has the most non-empty river-containing locations surrounding it (randomising for tie breaking).

3.3 Results

3.3.1 Observations

The example output of the graph generating algorithm shown in figure 13 appears reasonably successful in connecting rivers. However, the image tiles that contain blue are more concentrated in the central region of the mosaic that grows initially, rather than creating linear structures. For the image tiles with diagonal rivers diamond and cross patterns tend to occur, rather than continuous lines of river across the mosaic.

Figure 14 displays an additional run of the algorithm with seed set to 0, alongside the cell representation of each image in its arrangement to match the mosaic.

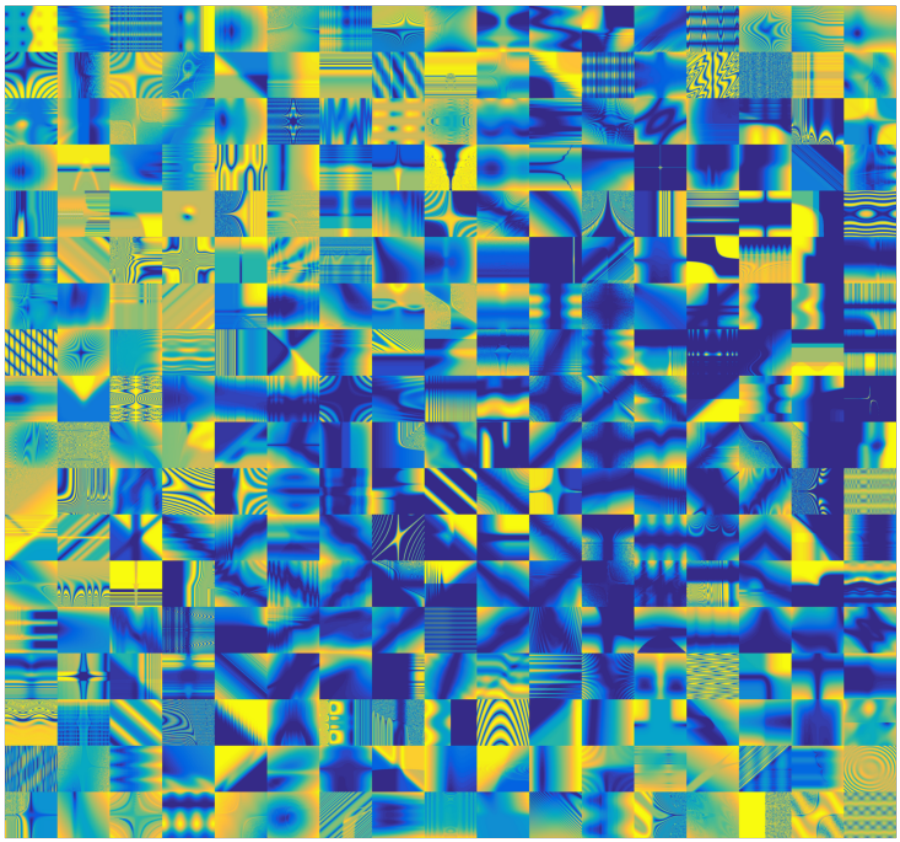
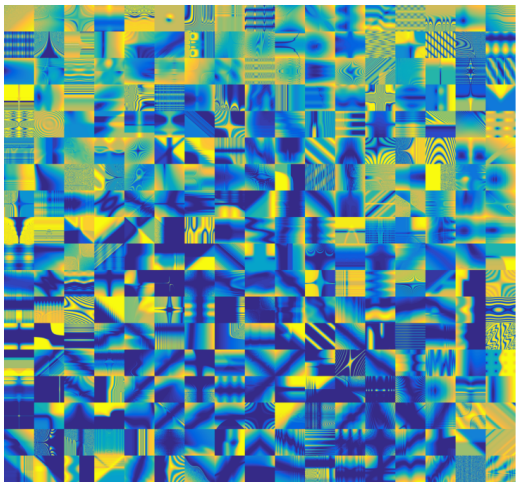


Figure 13: Maximising river connections with graph generating algorithm.



(a) Mosaic

(b) Cells

Figure 14: Comparison of a final mosaic with its cell representation

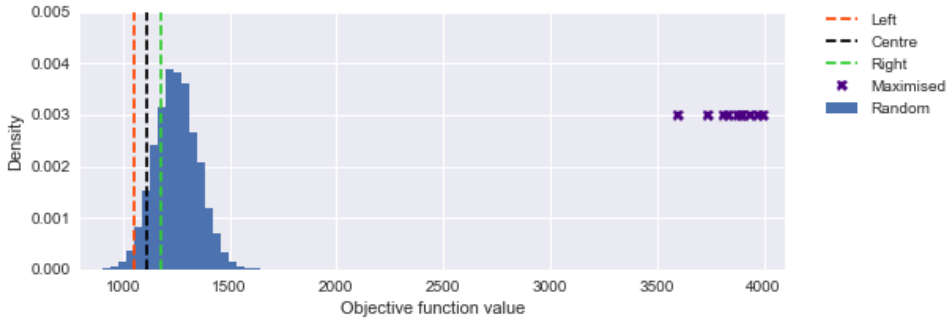


Figure 15: Results from 10 runs of graph generating algorithm

3.3.2 Analysis

The graph in figure 15 shows values of the objective function applied to the outputs of the graph generating algorithm (crosses), 10000 randomly generated mosaics (histogram), and the mosaics from the artwork (dashed lines). The final mosaics resulting from applying the graph generating algorithm scored consistently highly in comparison to randomised arrangements and the mosaics from the Negentropy Triptych.

3.3.3 Possible Improvements

Further development of the graph generating algorithm may include considering the aesthetic issues arising from the observation of the behaviour of the graph generating algorithm. These observations could be used to guide further modifications, refinement and extension of the algorithm. For example, choosing the next location in a directional manner that promotes linear structures or incorporating a measure of direction into the objective function.

Discussion and Conclusion

This project aimed to build on the previous work of Smith-Miles and Muñoz [1] by investigating methods of computationally generating the arrangement of visual representations of functions. We applied simulated annealing techniques to the previously developed greedy swap algorithm in an attempt to improve the formation and destruction of rivers. This approach was of limited success when applying visual assessment.

Continuing on, we developed a new approach using cell-based tile simplification and graph construction. This approach appeared slightly more successful in connecting rivers and forming larger groupings of blue, however, was not as successful in generating discrete linear forms of rivers as was seen in manual swapping.

Although it would be possible to further improve on the graph generating algorithm, other approaches may include machine learning and image processing techniques. Another possible approach may be to focus on matching edges of tiles, rather than assessing the full content of the tile.

While this research is focused on automated art generation, the potential applications of new methods for enhancing or destroying structure in images have important applications in image security and cryptography.

Acknowledgements

Thank you to Kate Smith-Miles and Andrés Muñoz for supervising this project. This work was supported by the AMSI Vacation Research Scholarship.

References

[1] Kate Smith-Miles and Mario Andrés Muñoz. "Human versus computer construction of mathematical artworks on an order-disorder aesthetic spectrum". In: *Journal of Mathematics and the Arts, submitted (under review)* (2021).

[2] Smith-Miles. *ARC Laureate project MATILDA*. URL: `http://www.matilda.unimelb.edu.au`.

[3] M. A. Muñoz and K. A. Smith-Miles. "Generating New Space-Filling Test Instances for Continuous Black-Box Optimization". In: *Evolutionary Computation* vol. 28. no. 3 (2020), pp. 379–404.

[4] M. A. Muñoz. *AUTOART: A greedy heuristic to construct mosaics from optimization function contour plots*. 2021.