

Distributed MAP-Elites and its Application in Evolutionary Design

Derek Hon

Submitted in partial fulfillment
of the requirements for the degree of

Master of Science

Department of Computer Science
Brock University
St. Catharines, Ontario

Abstract

Quality-Diversity search is the process of finding diverse solutions within the search space which do not sacrifice performance. MAP-Elites is a quality-diversity algorithm which measures n phenotypes/behaviours of a solution and places it into an n -dimensional hypercube based off its phenotype values. This thesis proposes an approach to addressing MAP-Elites' problem of exponential growth of hypercubes. The exponential growth of evaluation and computational time as the phenotypes/behaviours grow is potentially worse for optimization performance. The exponential growth in individuals results in the user being given too many candidate solutions at the end of processing. Therefore, MAP-Elites highlights diversity, but with the exponential growth, the said diversity is arguably impractical.

This research proposes an enhancement to MAP-Elites with Distributed island-model evolution. This will introduce a linear growth in population as well as a reasonable number of candidate solutions to consider. Each island consists of a two dimensional MAP which allows for a realistic analysis and visualization of these individuals. Since the system increases on a linear scale, and MAP-Elites on an exponential scale, high-dimensional problems will show an even greater decrease in total candidate solution counts, which aids in the realistic analysis of a run. This system will then be tested on procedural texture generation with multiple computer vision fitness functions. This Distributed MAP-Elites algorithm was tested against vanilla GP, island-model evolution, and traditional MAP-Elites on multiple fitness functions and target images. The proposed algorithm was found, at the very minimum, to be competitive in fitness to the other algorithms and in some cases outperformed them. On top of this performance, when visually observing the best solutions, the algorithm was found to have been able to produce visually interesting textures.

Acknowledgements

I would like to first thank Professor Brian J. Ross for his supervision, guidance, support, and belief over the years. It has made the graduate program a lifetime experience which I will remember for the rest of my life. I would then like to thank my parents for all their support and love they have given me. Next I would like to thank my friends, Liam McDevitt and Andrew Pozzuoli for being there with me for all the years I've known them and their support and comradery. Much thanks to the Faculty's Systems Administrator, Cale Fairchild, for all his assistance with the cluster and guidance throughout it.

Contents

List of Tables

List of Figures

1	Introduction	1
1.1	Proposed Solution	3
1.1.1	Diversity Search	3
1.1.2	Island-Model Evolution	5
1.1.3	Procedural Textures	6
1.2	Motivations and Goals	6
1.2.1	Motivations	6
1.2.2	Goals	8
1.3	Thesis Structure	8
2	Background	10
2.1	Genetic Programming	10
2.1.1	Initialization	11
2.1.2	Selection	12
2.1.3	Crossover and Mutation	12
2.2	Map-Elites	13
2.3	Island-Model Evolution	15
2.4	Procedural Textures	17
2.5	Image Features	18
2.5.1	Feature Definition	18
3	System and Experiment Design	21
3.1	GP System	21
3.2	MAP-Elites	24
3.3	Distributed GP and MAP-Elites	26

3.4	Procedural Texture Language	28
3.5	Direct Match	32
3.6	Wavelet Analysis	32
3.6.1	Haar Wavelets	32
3.6.2	Fitness Function	34
3.7	SSIM	37
3.8	CHISTQ	40
4	Procedural Texture Experiments	44
4.1	Direct Match Experiment	44
4.1.1	Vanilla GP Results	46
4.1.2	Island-Model Results	50
4.1.3	MAP-Elites Results	53
4.1.4	DME Results	56
4.1.5	Discussion	60
4.2	Wavelet Experiment	65
4.2.1	Vanilla GP Results: Barcode	67
4.2.2	Vanilla GP Results: Cartoon Smiling Face	70
4.2.3	Island-Model Results: Barcode	73
4.2.4	Island-Model Results: Cartoon Smiling Face	76
4.2.5	MAP-Elites Results: Barcode	79
4.2.6	ME Results: Cartoon Smiling Face	83
4.2.7	DME Results: Barcode	86
4.2.8	DME Results: Cartoon Smiling Face	90
4.2.9	Discussion	93
4.3	Wavelet SSIM Experiment	99
4.3.1	Vanilla GP Results: Wavelet	101
4.3.2	Vanilla GP Results: SSIM	104
4.3.3	Island-Model Results: Wavelet	107
4.3.4	Island-Model Results: SSIM	110
4.3.5	MAP-Elite Results: Wavelet	112
4.3.6	MAP-Elite Results: SSIM	116
4.3.7	DME Results: Wavelet	118
4.3.8	DME Results: SSIM	122
4.3.9	Discussion: Wavelet	125
4.3.10	Discussion: SSIM	129

4.4	Wavelet SSIM CHISTQ Experiment	132
4.4.1	Island-Model Results	133
4.4.2	DME Results	135
4.5	Discussion	138
4.6	Comments	141
5	Conclusion	144
5.1	Issues	145
5.1.1	High Level Issues	145
5.1.2	Low Level Issues	145
5.2	Future Work	147
	Bibliography	149
	Bibliography	154
	Appendices	155
A	Additional Experimental Analysis	155
A.1	Wavelet CHISTQ SSIM Experiment Run Results	156

List of Tables

2.1	Lombardi Feature Tests	19
3.1	GP Parameters	23
3.2	MAP Parameters	25
3.3	Distributed Parameters	27
3.4	GP Language	29
3.5	Procedural Texture Variables	30
3.6	Wavelet Value Comparisons	36
3.7	SSIM Variables	38
3.8	SSIM Value Comparisons	40
3.9	CHISTQ Variables	42
3.10	CHISTQ Value Comparisons	43
4.1	Fitness Function and Behaviours	45
4.2	GP Image Results	46
4.3	IM Image Results	50
4.4	Behaviour Intervals	53
4.5	ME Image Results	54
4.6	Direct Match DME Sub-MAPs	56
4.7	DME Image Results	57
4.8	Fitness Function and Behaviours	65
4.9	GP Image Results	67
4.10	GP Image Results	70
4.11	IM Image Results	73
4.12	IM Image Results	76
4.13	Behaviour Intervals	79
4.14	ME Image Results using Target Figure 4.17	80
4.15	ME Image Results using Target Figure 4.17	83

4.16	Wavelet DME Sub-MAPs	86
4.17	DME Image Results using Target Figure 4.17	87
4.18	DME Image Results using Target Figure 4.18	90
4.19	Fitness Function and Behaviours	100
4.20	GP Image Results	101
4.21	GP Image Results	104
4.22	IM Image Results	107
4.23	IM Image Results	110
4.24	ME Image Results using Target Figure 4.63	113
4.25	ME Image Results	116
4.26	DME Image Results using Target Figure 4.63	119
4.27	DME Image Results	122
4.28	Fitness Function and Behaviours	133
A.1	IM Subpopulation 1 Image Results	156
A.2	IM Subpopulation 2 Image Results	157
A.3	IM Subpopulation 3 Image Results	158
A.4	DME Sub-MAP 1 Image Results	159
A.5	DME Sub-MAP 2 Image Results	160
A.6	DME Sub-MAP 3 Image Results	161

List of Figures

2.1	Genetic Programming Tree For Expression: $\sqrt{2/x_1} + x^2$	11
2.2	Genetic Programming Reproduction	13
3.1	DME Island Configuration	28
3.2	Test Images	36
4.1	Target Image	44
4.2	Best Solution	46
4.3	Colour Channel Heat Maps	47
4.4	Best GP Fitness Graph	48
4.5	Best Solution	50
4.6	Island-Model Colour Channel Heat Maps	51
4.7	Best Fitness	52
4.8	Best Solution	54
4.9	Average Fitness	55
4.10	Best Solution	57
4.11	Direct Match DME Visualization	58
4.12	Best Fitness	59
4.13	Best Fitness Comparison	60
4.14	Behaviour plot using the best solutions from each algorithm	62
4.15	Tree Depth Box Plot	63
4.16	Critical Difference Diagram using best solutions from each algorithm	64
4.17	First Target	65
4.18	Second Target	65
4.19	Barcode Coefficients	66
4.20	Smile Coefficients	66
4.21	Best Solution	67
4.22	Best Solution Coefficients	67

4.23 Entropy Luminosity Heat Map	68
4.24 Best Fitness	69
4.25 Best Solution	70
4.26 Best Solution Coefficients	70
4.27 Entropy Luminosity Heat Map	71
4.28 Best Fitness	72
4.29 Best Solution	73
4.30 Best Solution Coefficients	73
4.31 Entropy Luminosity Heat Map	74
4.32 Best Fitness	75
4.33 Best Solution	76
4.34 Best Solution Coefficients	76
4.35 Entropy Luminosity Heat Map	77
4.36 Best Fitness	78
4.37 Best Solution	80
4.38 Best Solution Coefficients	80
4.39 Best MAP using Target Figure 4.17	81
4.40 ME Fitness using Target Figure 4.17	82
4.41 Best Solution	84
4.42 Best Solution Coefficients	84
4.43 Best MAP using Target Figure 4.18	84
4.44 ME Fitness using Target Figure 4.18	85
4.45 Best Solution	87
4.46 Best Solution Coefficient	87
4.47 Wavelet DME Visualization	88
4.48 DME Fitness	89
4.49 Best Solution	91
4.50 Best Solution Coefficient	91
4.51 Wavelet DME Visualization	91
4.52 DME Fitness	92
4.53 Barcode Fitness Comparison	93
4.54 Barcode Luminosity Box Plot	94
4.55 Barcode Entropy Box Plot	94
4.56 Barcode Tree Depth Box Plot	94
4.57 Barcode Critical Difference	95
4.58 Cartoon Smiling Face Fitness Comparison	96

4.59	Luminosity Box Plot	97
4.60	Entropy Box Plot	97
4.61	Cartoon Smiling Face Tree Depth Box Plot	97
4.62	Cartoon Smiling Face Critical Difference	98
4.63	Target Image	99
4.64	Sunset Coefficients	99
4.65	Best Solution	101
4.66	Best Solution Coefficient	101
4.67	Entropy Luminosity Heat Map	102
4.68	GP Fitness	103
4.69	Best Solution	104
4.70	Entropy Luminosity Heat Map	105
4.71	GP Fitness	106
4.72	Best Solution	107
4.73	Best Solution Coefficient	107
4.74	Entropy Luminosity Heat Map	108
4.75	IM Fitness	109
4.76	Best Solution	110
4.77	Entropy Luminosity Heat Map	111
4.78	IM Fitness	112
4.79	Best Solution	113
4.80	Best Solution Coefficient	113
4.81	MAP Visualization	114
4.82	GP Fitness	115
4.83	Best Solution	117
4.84	SSIM ME Visualization	117
4.85	ME Fitness	118
4.86	Best Solution	119
4.87	Best Solution Coefficient	119
4.88	Wavelet DME Visualization	120
4.89	DME Fitness	121
4.90	Best Solution	123
4.91	SSIM DME Visualization	123
4.92	DME Fitness	124
4.93	Wavelet Fitness Comparison	125
4.94	Luminosity Box Plot	126

4.95 Entropy Box Plot	126
4.96 Tree Depth Plot	127
4.97 Critical Difference Diagram using best solutions from each algorithm	128
4.98 Wavelet Fitness Comparison	129
4.99 Luminosity Box Plot	130
4.100 Entropy Box Plot	130
4.101 Tree Depth Plot	131
4.102 Critical Difference Diagram using best solutions from each algorithm	132
4.103 Best Subpopulation Solutions	133
4.104 Entropy Luminosity Heat Map	134
4.105 IM Fitness	135
4.107 Best Overall Sub-MAP configuration	136
4.106 Best Subpopulation Solutions	136
4.108 DME Fitness	138
4.109 Fitness Comparison	139
4.110 Entropy and Luminosity Behaviour Box Plots	140
4.111 Tree Depth Box Plot	140

List of Algorithms

1	Genetic Algorithm Pseudocode	11
2	Map-Elites Pseudocode	14
3	Procedural Texture Algorithm	31
4	Direct Match Algorithm	32
5	Haar Wavelet Decomposition	33
6	Haar Wavelet Truncated Fitness Function	35
7	SSIM Fitness Calculation	39
8	CHISTQ fitness calculation	42

Chapter 1

Introduction

Evolutionary design is a field that looks into the world of creativity and explores this space through evolving solutions. For example, procedural texture generation [15] is a popular application within this field. Within computer graphics, procedural textures provide a texture that requires little memory due to being generated by an algorithm [41]. Due to this, the texture also has no set resolution and any such resolution can be generated. The algorithm can use the x and y coordinates to generate the colour at that specific coordinate point.

Diversity is a key term in evolutionary design which denotes the difference in solutions generated. A wide range of art pieces in different styles is considered diverse. Having a wide range of art makes things interesting and enables the viewer to appreciate each individual style more, depending on their subjective own tastes. Without diversity in art, everyone would be producing the exact same thing for eternity. Diversity is what enables evolution to create different solutions, by having a diverse set of solutions in the pool it will continue to try new possibilities.

Diversity search focuses on diversity, rather than exclusively on optimization. In typical optimization, the evolution focuses on providing the best fitness value possible, whether minimizing or maximizing. Diversity search tries to find different solutions where possible. An example of this is novelty search, where the novelty of a solution

is desired [29]. The novelty of a solution is determined by how different it is than other solutions. The value in this is that not all desired solutions can be defined adequately by a fitness function alone. For example, a common deceptive problem is maze navigation where an agent is to navigate a maze and reach an exact point. Optimization algorithms on maze problems will typically fail if "distance to the exit" is used as the fitness. However, if a novelty search of "go to a new unexplored part of the maze" is used, solutions can be found. Similar to mazes, art and design cannot always be defined by a simple fitness function. Two people observing the same architecture or painting can have completely different opinions on it, which is why novelty is important. Novelty in evolutionary design can produce many different solutions which may contain something the developer is specifically looking for which the fitness function cannot quantify.

Related to novelty search, quality-diversity (QD) search [45] aims to find solutions within the search space which are as diverse as possible with respect to the behaviours within the space, while also maintaining high performance. To summarize the direct comparison between the two, novelty compares solutions to one another where QD focuses on diverse areas within the search space whilst performing well in regards to the fitness value.

The QD search algorithm, Multidimensional Archive of Phenotypic Elites (MAP-Elites) tracks certain aspects, or behaviours, of a solution and keeps the best performing solutions with the measured behaviours [38]. One problem is that measuring too many behaviours results in too many possible solutions. For example, if there are 3 behaviours with 7 intervals for each behaviour this will result in $7^3 = 343$ possible solutions.

island-model (IM) evolution is a technique utilized within Genetic Algorithms (GA) and Genetic Programming (GP) which increase genetic diversity through the usage of multiple populations communicating to one another. This is a possible

compromise for the issue of many behaviours producing too many candidate solutions.

1.1 Proposed Solution

This thesis proposes a new MAP-Elites island-model system as one possible answer to the problem of many variables with varying complexity. This system creates multiple sets of solutions where each set can optimize for slightly different things. These sets of solutions will also be communicating with each other, taken from island-model evolution, and transferring solutions in order to provide and promote more diversity within each set. This is an exponential growth in terms of the number of behaviours. Besides being potentially harmful in algorithmic performance, it can also result in supplying too many solutions to the user. These two diversity techniques, MAP-Elites and island-model evolution, produce diversity in structurally different ways which allow for combination to be possible. MAP-Elites works based on the optimization of multiple features, where island-model provides multiple sets or populations and the communication between sets which is inspired from parallel genetic algorithms [38][7].

1.1.1 Diversity Search

Lehman and Stanley proposed an technique within evolutionary computation called novelty search [30]. This search technique learns through rewarding functionally different solutions rather than searching for a final objective.

Vassiliades *et al.* [51] acknowledge the main drawback of MAP-Elites being the inability to scale into high dimensional feature spaces due to exponential growth. They utilize computational geometry to partition the high-dimensional space into well-spread geometric regions. Their algorithm uses centroidal Voronoi tessellation (CVT) to divide the feature space into x regions. Individuals are placed into the

closest region. CVT-Map-Elites was found to scale much better than MAP-Elites.

Basher applies a variant of MAP-Elites to procedural texture synthesis [5]. He proposes a new MAP-Elites algorithm that focuses on the cells within the grid. He uses 2-D MAP-Elites with many-objective fitness optimization, in other words, many fitness scores must be optimized simultaneously. His system shares similar optimization performance to vanilla GP. Also, each cell is considered a bin that can hold N individuals, rather than only 1. By increasing the potential individuals with the same behaviour features it is hoped to increase diversity.

Alvarez *et al.* [2] successfully created diverse designs of dungeons within procedural content generation utilizing MAP-elites and its illumination abilities.

Khalifa *et al.* [26] utilized MAP-elites for a different kind of content generation, specifically level generations for action games that require avoiding extreme amounts of projectiles. This framework combines MAP-elites with Feasible-Infeasible 2-Population Genetic Algorithm which resulted in successful creation of levels of varying difficulty.

Fontaine *et al.* [17] devised a modification on MAP-Elites, MAP-Elites with Sliding Boundaries (MESB), which would design and build decks for the strategic online card game, Hearthstone. The sliding boundaries comes from setting boundaries at percentages on the distribution rather than at specific feature values allowing for more flexibility. This research found that the modification on MAP-Elites played Hearthstone in diverse and advantageous ways while also revealing a common patterns that recur through dimensions.

Colas *et al.* [10] addressed MAP-Elites limitation of low-dimension controllers within robotics. They propose a system utilizing Evolution Strategies to scale MAP-Elites to high-dimensional controllers parameterized by large neural networks. They prove that ME-ES performs efficient exploration competitively with state-of-the-art exploration algorithms.

Other examples of MAP-Elites research includes uses in GP program representation [13], bipedal walkers [25], noise [16], and gait controllers [39].

1.1.2 Island-Model Evolution

Island-Model evolution is the concept where a population is subdivided into smaller subpopulations and these islands exchange genes with other islands [28]. Island-model evolution takes this concept and applies it to parallel GAs [8]. Rather than genes, each island sends solutions to other islands at the discretion of the user. The segregated islands promote diversity through their own evolutions resulting in potentially different solutions and then migrating these said solutions. Koza and Andre [3] found that using this to parallelize GPs resulted in super linear speed-ups in computational effort.

Research utilizing island-model evolution includes the combination of age-layering with novelty search with IM [37], processing linearly separable algorithms [56], parameter tuning for benchmark functions [19], optimizing Artificial Neural Network parameters and weights [23], soft topologies [1], differential evolution with an immigration pool [32], dynamic migration policy [14], lack of migration on discrete-continuous scheduling with continuous resource discretization [49], and specific region searching in GP [11].

1.1.3 Procedural Textures

There is a lot of research that explores GP on the domain of evolutionary design and procedural textures [58, 18, 46, 22, 9, 20, 21, 34]. One example in the Gentropy system, an island-model multi objective GP system that would evolve 2-D procedural textures. The three fitness functions utilized within the topology are: colour histogram quadratic matching (CHISTQ), wavelet analysis (WAV), and smoothness histogram matching (SHIST). Some sub populations would only calculate the fitness using one of these functions, some would take two functions and have a weighted sum of half and half, and the last variation would take the three multiplied by 0.33 for an even split. This setup provided success in automatically generating textures, and its user-selectable image analysis fitness functions proved to be effective when automating aesthetics [57].

1.2 Motivations and Goals

1.2.1 Motivations

This research provides a new approach towards multi-behaviour diversity in evolution through the combination of island-model evolution and MAP-Elites. The motivation behind this approach is to provide higher dimensionality for behavioural diversity while minimizing computational overhead. MAP-Elites allows for a multi-dimensional archive, or map of solutions. However, there exists some problems with this. If a map contains k number of behaviours it becomes a hypercube of k dimensions. If each behaviour has a resolution of m behaviour categories (cells within the map), then the total hypercube will have m^k cells. This is a problem because the map will grow exponentially with the number of dimensions. This provides a sparse and dispersed population which may result in less effective optimization. When looking

at the end-of-run solutions, it then becomes increasingly difficult to observe due to the sheer number of individuals to consider.

Island-Model MAP-Elites provides a smaller overall population. This is because it only utilizes 2-D maps, since each map has 2 behaviours. If each behaviour has m categories, a map would then have a total of m^2 cells under the assumption of equal categories per behaviour. There would then be multiple islands of these 2x2 maps that would communicate with each other.

For example, consider $k = 3$ for behaviours A, B, and C we have 3 distinct maps: AB, AC, and BC. If each map is of size 10x10, then there are $3 \times 10 \times 10 = 300$ solutions in the overall population. A 3-dimensional (hyper)cube is $10^3 = 1000$ overall solutions. This is a reduction of 70% as 300 is 30% of 1000. If raised to $k = 4$ behaviours, this results in 6 distinct pairs. MAP-Elites would produce $10^4 = 10,000$, whereas DME would produce $6 \times 10 = 600$. There is a difference of 94% of solutions since 600 is 6% of 10,000. The island's reduced size is even more pronounced with a greater number of behaviours. The hope is that the proposed island-model MAP-Elites system is much more practical than multi-dimensional maps in terms of space as well as time, while also providing more diverse results in comparison to vanilla evolution. It should be more effective for multiple behaviours assuming better convergence while also supplying fewer final solutions to the user.

The purpose behind applying this system to evolutionary art is because diversity and subjectivity are extremely important when it comes to artistic creative applications. Diversity can be quantified with the usage of multiple behaviours and the number of intervals for each behaviour. However, subjectivity cannot be quantified because each person will perceive things differently. Art is an inherently subjective field that cannot be optimized due to emotion, and the differences in each person's senses. Creativity and expression is individualistic and is moulded by a person's experiences which is why it cannot be boiled down to a single or multiple fitness

functions. Due to this subjectivity it is important to provide a diverse set of solutions to the user and have them observe each solution. This system is not only applicable evolutionary design but other problem domains like intelligent agents [44]. Andrea Wiens' Gentropy [57] was very inspirational in this work being in the same problem domain of evolving procedural textures while it explores diversity. Wiens explored diversity through different island configurations while this research is concerned with MAP-Elites.

1.2.2 Goals

1. Add multiple diversity behaviours to an application without the exponential increase seen with high-dimensional MAP-Elite hypercubes.
2. Compare diversity and quality of solutions given by Distributed MAP-Elites to the baseline runs of MAP-elites, island-model evolution, and GP.
3. Discover a practical strategy to examine multiple solutions from at least one run from the Distributed MAP-elites
4. Test Distributed MAP-elites on an evolutionary design application (procedural texture synthesis).

1.3 Thesis Structure

The thesis is structured as follows:

- Chapter 2 provides background information with regards to concepts related to this thesis: MAP-Elites, island-model, Procedural Textures.
- Chapter 3 explains the system design: the algorithm, the applications to be used, and the feature space.

- Chapter 4 highlights the results and discussion for each experiment conducted.
- Chapter 5 summarizes the thesis and discusses future work.

Chapter 2

Background

2.1 Genetic Programming

Genetic programming (GP) is a computational intelligence technique which involves evolving programs to solve specific problems [27] [42]. These programs are evolved using the same natural selection concepts as found in GAs. The individuals, in this case programs, are evaluated for fitness to determine how well they solve the problem. Reproduction that produces offspring follow the same concepts of crossover and mutation. However the way in which these operations are performed are different. This is because GP operates with tree-based chromosomes. Algorithm 1 illustrates aspects of GAs which are core to the algorithm.

```

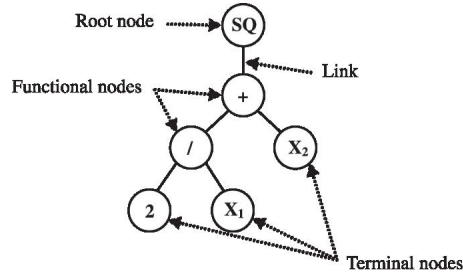
[1] Initialize x number of individuals;
while not maximum generations do
    [2] Choose two individuals;
    [3] Breed two new individuals;
    [4] Evaluate the two new individuals;
    if Ideal solution is found(Optional) then
        [5] Exit loop;
    end
end

```

Algorithm 1: Genetic Algorithm Pseudocode

2.1.1 Initialization

There is a population of individuals that the algorithms evolve. This population must be initialized with individuals in order to perform evolution.

Figure 2.1: Genetic Programming Tree For Expression: $\sqrt{(2/x_1) + x_2^2}$

The tree-based chromosomes in GP get randomly generated with function nodes and terminal nodes. The root node determines the value of the individual which can be used in different ways depending on the problem. Figure 2.1 shows the different parts of the tree. Functional nodes are operators that take in operands, but not necessarily multiple operands. There are operations like \log_{10} where only one operand is necessary. Operands of these functional nodes can be both functional and terminal

nodes as in Figure 2.1. The result of the division operator is used as one operand for the addition operator. Terminal nodes are static values that do not take any children or operands. These terminal nodes can be variables which the user passes in, x_1 and x_2 , or they can be values known as ephemeral random constants (ERC). ERCs are randomly initialized constants which persist in value for the rest of the run.

Within a tree are sub-trees, for example, Figure 2.1 contains a sub-tree of the division operator, the constant 2, and x_1 . Since these are randomly generated in this initialization phase, there are certain constraints. The depth of the sub-tree generated is controlled through minimum and maximum size parameters in order to limit the depth of the tree.

2.1.2 Selection

After the population of individuals has been initialized, the first step of evolution is to do fitness-based selection of parents to breed children. A common technique of selecting individuals is tournament selection. First, a number amount of individuals, typically 3-7, are randomly selected from the population. The fittest individual from this set is designated to be a parent. This process is then repeated for deciding the second parent for crossover. Mutation does not require two individuals and thus only uses tournament selection once to mutate said individual.

2.1.3 Crossover and Mutation

The third step highlighted in algorithm 1 is the breeding process. This breeding process involves crossover and mutation of the parents in order to create the children.

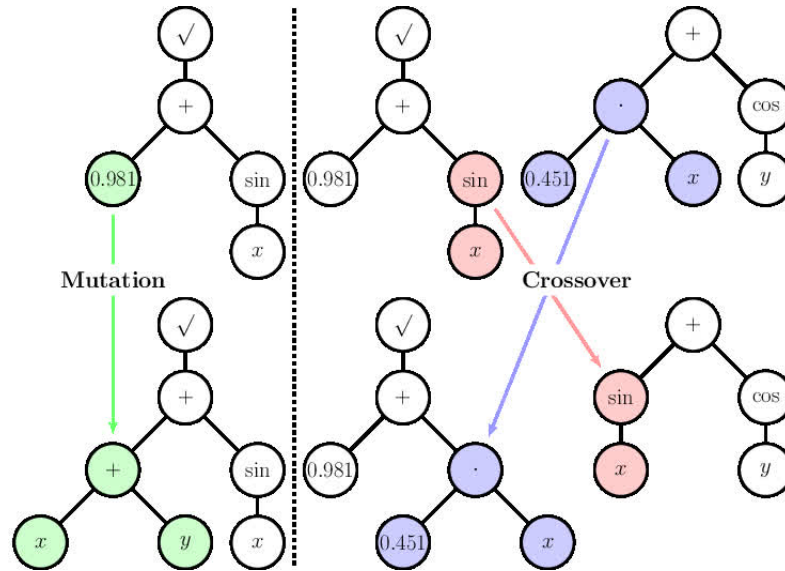


Figure 2.2: Genetic Programming Reproduction

The process of crossover is: randomly select a node on each parent tree, swap these two sub-trees, then the resulting trees are made into children for the next generation. Sub-trees are chosen to cross over with other sub-trees to ensure that trees are always syntactically correct (Figure 2.2).

Mutation begins by first determining a sub-tree to delete. This sub-tree is then replaced with a newly generated sub-tree which has been randomly generated.

2.2 Map-Elites

Map-Elites is an algorithm devised by Mouret and Clune [38]. Rather than an optimization algorithm, which would try to find the best performing solution within the search space, this is an illumination algorithm which tries to find the best performing solutions within each user-designated area of the feature space. Since it explores different aspects of the feature space, it is tailored towards diversity rather than optimization. Map-Elites provides an n -dimensional map to house individual solutions. Each of the n dimensions is a feature, and the solutions would have a behaviour value tied to it in order to be placed within the map. Each dimension or behaviour has

intervals which are set to the user. For example, there may be a behaviour that measures the average red channel of an image. The intervals set for the behaviour can be 0-100, 101-200, 201-255. Solutions with average red values within each interval would then be placed in those cells. Assuming k number of intervals for a behaviour or dimension, a 2-dimensional map would result in a $k \times k$ sized grid.

```
[1] Initialize a number of individuals or percentage of total possible solutions ;
while not maximum evaluations do
|
| [2] Choose two random individuals;
| [3] Breed new individual;
| [4] Generate behaviour/phenotype values for new individual;
| [5] Obtain cell in the MAP for these behaviour values;
| if cell empty then
| | [6] Place individual inside cell;
| else
| | [7] Replace individual if new individual fitness is better;
| | [8] Replace individual if the new individual is equivalent in fitness and
| |     passes replacement probability;
| end
end
```

Algorithm 2: Map-Elites Pseudocode

Algorithm 2 is the pseudocode for the Map-Elites algorithm when evolving individuals. After generating a children solution or solutions, the feature values or behaviours would be evaluated for the solution in order to place it within its respective cell within the map based on the aforementioned intervals. If that cell is already filled by another solution then a fitness comparison is done so that the more fit solution is placed within the cell.

2.3 Island-Model Evolution

Island-model evolution is a different way to achieve diversity in evolutionary algorithms [7]. It is also a classical approach to parallelism within evolutionary algorithms. Island-model evolution is comprised of N number of subpopulations, also referred to as islands or demes, as decided by the user. Each island is an independent subpopulation that evolves on its own most of the time. However, at regular intervals, islands "migrate" individuals to each other. The result is, when successful, a distributed evolutionary algorithm that evolves better solutions faster than a single population.

Island-model evolution brings along many new parameters with the existence of subpopulations:

1. Number of subpopulations
2. Individual subpopulation parameters:
 - (a) Subpopulation Size
 - (b) Breeding parameters (Crossover/Mutation etc.)
 - (c) Fitness function
3. Island/Subpopulation topology:
 - (a) Selection method
 - (b) Migration interval
 - (c) Generation to start migration
 - (d) Number of individuals to migrate
 - (e) Destinations to migrate

Each subpopulation is an instance of a population which means that they all need their own population GP parameters. It is necessary to state the number of subpopulations so GP knows how many subpopulation parameters it needs to process. As

previously stated, these subpopulations require their own population GP parameters which are identified under "Individual subpopulation parameters". They are capable of processing individuals in separate ways thanks to the fitness function, however, it is also possible to maintain all subpopulations with the same fitness function. This allows for a new outlook to diversity, by trying to optimize for different combinations of fitness functions connected through migration. As for the topology, the system needs a way to choose individuals to migrate over to different subpopulations, which is why each subpopulation will define a selection method for migration. The same for the following parameters under "Island/Subpopulation topology". It is possible to have the different subpopulations migrate individuals at different rates which is defined by generation count, for example, a rate of 5 would mean every 5 generations a migration would occur to the designated destinations. The number of individuals to migrate remains consistent for the destinations. To elaborate on this, if the number of individuals to migrate is 10, and there are 5 destinations, it will send 10 individuals to each of the 5 destinations. The destination refers to the specific subpopulation(s) it is migrating individuals to.

Island-model may produce superior search performance to that of single population models because of the linear separability of a problem by decomposing a problem into subproblems and solving these subproblems separately before building the full answer through migration. Whitley [55] devised a model that hints to the fact that island-model may be advantageous when increasing the population size no longer helps solve the problem, however they acknowledge there are many variables unexplored to this model. However, the diversity maintenance and the linear separable nature of it does prove that it has the ability to solve some problems better than that of single population models. Koza and Andre [3] successfully introduce a parallel implementation of GP that achieves a super linear speed-up with less computational effort. The two speed-ups identified are the near linear speed-up in code execution

and more than linear speed-up in solving the problem.

2.4 Procedural Textures

Procedural textures are a procedurally generated images using an algorithm or formulae. They contrast to storing photographic pixel data of an image into memory [54]. Since the memory now only holds an algorithm to generate the texture as needed, there is vastly more free memory with this method. The main advantage to using procedural textures is their ability to compute complex designs and patterns at any resolution whilst avoiding tiling in bitmaps. This also makes texture mapping easier [15]. Here is an example of a procedural texture algorithm:

$$R(x, y) = \sin(x) \tag{2.1}$$

$$G(x, y) = \cos(y)\sin(x) \tag{2.2}$$

$$B(x, y) = x - y \tag{2.3}$$

Utilizing the x and y coordinates, red, green, and blue values are calculated through their respective formulas and thus form the RGB value for that specific pixel.

Procedural textures do come with some problems, they can be computationally expensive especially if large algorithms and formulae are used. Secondly, they can be difficult to hand-design to create desired effects. Often, well known procedural textures are parameterized for users; for example, Perlin noise [40] is commonly used for realistic stone effects. Procedural textures produce colour values where texture images created are defined over a continuous range of texture coordinates and thus is the baseline for high-quality "shader-based" texturing approaches [36].

2.5 Image Features

This section will highlight image features that are used in the experiments pertaining to procedural textures. The majority of the features were obtained from Lombardi *et al.*'s set [31]. This set of features are lightweight meaning they are efficient to compute

2.5.1 Feature Definition

Lombardi et al. [31] provided the set of features through their model that described and analyzed digitized paintings. Two principle features are defined by the model, palette and canvas. The palette refers to the set of colours required to make a painting where they state is derived from the colour map of an image. Canvas features focus on frequency and spacial distribution of these colours.

Within their research, they developed two feature sets to test their system. The first feature set works within the red, green, and blue (RGB) colour space. This set utilizes one palette feature and fifteen canvas features. The second feature set operates within the hue saturation value (HSV) colour space which corresponds more towards human perception than that of the RGB colour space. The second set is comprised of eighteen canvas features.

Feature Name	Type	Description and Notes
Red Mean	Canvas	The arithmetic mean of values in the R channel
Green Mean	Canvas	The arithmetic mean of values in the R channel
Blue Mean	Canvas	The arithmetic mean of values in the B channel
Intensity Mean	Canvas	The global brightness of an image
Colour Entropy	Canvas	The degree of disorder in the frequency distribution of colours

Table 2.1: Lombardi Feature Tests

Table 2.1 are the features which have been utilized in this thesis. Intensity, mean, and colour entropy are canvas features which are a part of the second feature set, while the others are part of the first preliminary feature set.

Unique triplets refer to the number of unique pixel value combinations in an image. Poynton [43] illustrates that colour images are generally best captured at the red, green and blue spectrums. The combination of these three channels will result in the colours that people are familiar with. The RGB triplet have values that range from 0 to 255 giving 256^3 combinations of colours. However, the human eye is not able to discern the subtle difference between all these colour values.

Poynton[43] points towards the important component of lightness in video systems conveying data. Lightness is conveyed in a perceptually uniform manner that minimizes the amount of noise or quantization error. The weighted sum of the respective coefficients and the colour channels of red, green, and blue would form a luma(Y') signal representative of lightness. An example of this is the International Telecommunication Union's recommendation for standard definition content:

$$Y' = 0.299R' + 0.587G' + 0.114B' \quad (2.4)$$

Within information theory, the entropy of a random variable is the average uncertainty of the possible outcomes. Shannon identifies three properties that this measure, $H(p_1, p_2, \dots, p_n)$, must adhere to [48]:

1. H should be continuous in the p_i
2. If all the p_i are equal, $p_i = \frac{1}{n}$, then H should be a monotonic increasing function of n . With equally likely events there is more than one choice, or uncertainty, when there are more possible events
3. If a choice be broken down into two successive choices, the original H should be the weighted sum of the individual values of H .

The only H that satisfies these three properties is:

$$H = -K \sum_{i=1}^n p_i \log p_i \quad (2.5)$$

Since an image can be described by each pixel's values, Shannon entropy can be applied to an image. The description used within this research will be the grayscale values rather than the unique colour triplet of red, green, and blue. Both unique colour triplets and grayscale values operate within the same range thus providing the ability to perform the same entropy calculation if either were to be used. Shannon entropy on images is very popular within computer vision.

Chapter 3

System and Experiment Design

This chapter discusses the system, Distributed MAP-Elites. It will provide details about the system’s implementation, parameters which are shared through experiments, the platform it is implemented on, the application it is applied to, and external tools utilized for the application. The systems that will be compared and contrasted to DME in this work are vanilla GP, island-model, and MAP-Elites. The following sections will outline parameters for all these systems.

3.1 GP System

This system will run on ECJ version 27 [33]. Evolutionary Computation Journal(ECJ) is a research EC system written in Java. ECJ evolves GP trees [27]. The reasoning for choosing ECJ was due to previous familiarity with the system, in addition to the many features it houses. There exists a steady state evolution application within ECJ that aided with the implementation of MAP-Elites, since the general concept using a steady-state population is similar. Steady-state evolution is where the population does not get replaced and does not run on a per-generation basis. As children are created and evaluated they will replace the worst performing individuals if they are better than said worst performing individuals. As for island-model, it is by default incorporated into the GP implementation of ECJ which provides a straight-

forward implementation for the distribution of DME. The GP system requires some basic parameters in order to run which are highlighted in Table 3.1. These are native to GP evolution and ECJ's Koza parameters [33].

PARAMETER	VALUE	DESCRIPTION
Evaluations/ Generations	300000/200	Total number of evaluations/generations in a single run
Population Size (GP/IM)	1500	Number of individuals in the population
Tree Initializer	Ramped Half-n-Half	Method that is used to create trees in the beginning
Initial Tree Depth	Min: 2 Max: 6	Minimum and maximum depths when creating trees
Crossover Rate	0.9	Probability of using the crossover operator for reproduction
Mutation Rate	0.3	Probability of using the mutation operator for reproduction
Crossover Max-Depth	17	Maximum depth to perform the crossover operator
Mutation Max-Depth	17	Maximum depth to perform the mutation operator
Retries	1	Number of retries when performing mutation or crossover
Selection Method	Tournament(GP/IM) Random (ME/DME)	Method that individuals are selected for breeding
Tournament Size (GP/IM)	3	Number of individuals selected for the tournament selection method

Table 3.1: GP Parameters

The evaluations/generations in Table 3.1 is what the GP uses to know how long to run for. Each evaluation is a singular individual being bred. A generation is when a

population of individuals are bred which is only evident in population based evolution. Vanilla GP utilizes a population of 1500, and island-model 1500 total individuals within its configuration. These algorithms are conducted on 200 generations. An evaluation is defined as the act of reproduction, thus, 1 generation would result in 1500 evaluations. Therefore, MAP-Elite variants will equivalently use $1500 \times 200 = 300,000$ evaluations. The crossover rate determines when parents will perform a crossover operation on the children. The mutation rate determines when a child solution performs the mutation operation. When performing crossover and mutation, there can be a limit to the tree where the system will not allow crossover and mutation to occur. If there is a node below a tree depth of 17 then that will not be allowed which results in a retry. Parents must be selected before breeding children and performing these operations. The selection methods used are random and tournament. Random does not require any parameters as it chooses randomly from the set or population. This is used within the MAP-Elite variant algorithms. Tournament will be used for the island-model evolution and vanilla GP. Tournament selection chooses from a subset of individuals which have randomly been selected from the total set. This subset or pool is the tournament size which is 3.

3.2 MAP-Elites

MAP-Elites, being a fundamentally different algorithm than vanilla GP, results in a unique set of parameters necessary to run it. There are some fundamental parameters that will not change between experiments whereas others highlighted will only be a boilerplate having the experiments detail specifically what they use.

PARAMETER	VALUE	DESCRIPTION
Starting Individuals	0.6	Percentage of the MAP to try and fill initially
Replacement Probability	1.0	How often to replace individuals with the same fitness
Behaviours	3	Number of behaviours to evaluate
Behaviour Names	Behaviour 1: Red Behaviour 2: Green Behaviour 3: Blue	Names for each behaviour
Behaviour Intervals	Behaviour 1: 3 Behaviour 2: 3 Behaviour 3: 3	Number of intervals for each behaviour
Interval Upper Bounds	Interval 1: 100 Interval 2: 200 Interval 3: 255	Upper bounds for each interval

Table 3.2: MAP Parameters

The starting individuals parameter in Table 3.2 will remain consistent throughout all experiments utilizing MAP-Elites variants. It will try to fill 60% of the total population of the MAP, otherwise it will evaluate until 10 times the maximum occupancy has been fulfilled. The replacement probability is there for cases when an individual is generated with the same fitness as an existing elite occupying the cell which it would be placed in. This probability will determine whether it replaces the pre-existing elite or not. The number of behaviours to be evaluated from a solution must first be set before the details, in this case it will be 3. 3 behaviours, or any behaviour count greater than 2, is considered high-dimensional due to the inability to easily vi-

sualize the proceeding results. Each behaviour will have a name denominating what it is evaluating, and it will have intervals to determine which cell on the MAP the solution should go in. In this example, there will be 3 intervals for each behaviour and each interval set will be the same for each behaviour. The interval upper bounds parameter delineates where the interval ends and where the next interval starts.

3.3 Distributed GP and MAP-Elites

Distributed MAP-Elites combines MAP-Elites and island-model distribution. Each island within the model is replaced with a MAP of elites instead of a population. The communication between the islands will stay the same where individuals are emigrated to other islands after a certain number of evaluations have passed. Each of these MAPs are their own entities and thus have individually defined behaviours and fitnesses. The MAPs sizes are fixed to two dimensions and thus will only have two behaviour measures for each island. Based off the total number of behaviours being considered, at the very least, there will be as many islands as there are unique pairs. However, the user is still capable of picking and choosing specific behaviours to explore relationships. The MAP requires some base parameters for during the initialization phase which will remain consistent. Islands in distributed GP, or island-model evolution, need to disclose their population size as opposed to each sub-MAP disclosing the number of behaviours and intervals. These two distributions will be utilizing the same fitnesses for all islands unless stated otherwise.

Other population specific parameters in Tables 3.1 and 3.2 are all the same for their respective algorithms.

PARAMETER	VALUE	DESCRIPTION
Sub-MAPs/Subpopulations	3	Number of islands in the distribution
Subpopulation Size (IM)	500	Size of subpopulation
Emigration Rate	4 generations 6000 Evaluations	Evaluations or generations to emigrate individuals
Start Offset	0	Number of generations or evaluations to pass before the first emigration
Emigration Size	5	Number of individuals to emigrate
Number of Destinations	1	Number of destinations for emigration
Destinations	Island 1->2 Island 2->3 Island 3->1	Destinations for each island

Table 3.3: Distributed Parameters

Every experiment involving a distributed algorithm will be utilizing 3 islands in its distribution. The island-model evolution experiments will maintain the 500 population size for each subpopulation to remain consistent with the 1500 individuals in the vanilla GP experiments. An emigration rate of 4 was chosen for all the islands in order to keep it simple, as well as having a large effect on the diversity through the frequent shuffling of individuals. The offset was not set as it is acceptable for the 4th generation to begin emigration and immigration. Each island will only send individuals to one other island to avoid complicating the architecture. The islands are set to send individuals to the subsequent island in a circle-like architecture, once again for simplicity.

Figure 3.1 is an example of the circle-like architecture. Each island will be sending 5 individuals from its own map to the next map. The individuals do not simply get

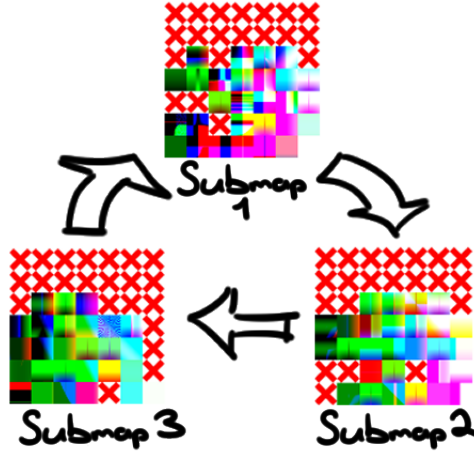


Figure 3.1: DME Island Configuration

placed into the map after being emigrated. For example, sub-map 2 would reevaluate the behaviours of the individuals sent by sub-map 1 to see which cell it would be placed in. If the cell it's supposed to go in is empty then it will be placed in there, otherwise a fitness comparison will be done and the fitter individual will take the cell. This is repeated for all individuals emigrated to sub-map 2. Each sub-map would perform this set of actions each time migration occurs.

3.4 Procedural Texture Language

Procedural textures reside in an infinite texture space(plane) and it must be specified which coordinates to use when generating the texture. The experiments conducted here will not explore different coordinates for the application's simplicity case. The minimum and maximum X and Y values used are -1 to 1.

Table 3.4 is the procedural texture language that makes up the solutions in the GP tree [18, 5]. This is a combination of mathematical functions, ERCs and problem specific values. The problem specific values, the coordinates of the image, are necessary in creating images. The texture coordinates must be able to be passed into the tree in order to generate a colour value for each pixel. The ERC values are there in

order to provide constant variance in each tree.

Table 3.4: GP Language

Function	Description
Add(x,y)	Adding two inputs together
Sub(x,y)	Subtracting the second input from the first input
Mul(x,y)	Multiplying two inputs together
Div(x,y)	Protected division between two inputs
Sin(x)	Sine function of an input
Cos(x)	Cosine function of an input
Pow(x)	Taking the first input as the base and second as the power
Log10(x)	Taking the protected logarithm base 10 of an input
Eph_1	Ephemeral value between 0 and 1
Eph_10	Ephemeral value between 0 and 10
Eph_100	Ephemeral value between 0 and 100
Pos_X	The X texture coordinate of the image
Pos_Y	The Y texture coordinate of the image

Table 3.5: Procedural Texture Variables

Variable	Description
Minimum X	Minimum X texture coordinate
Maximum X	Maximum X texture coordinate
Minimum Y	Minimum Y texture coordinate
Maximum Y	Maximum Y texture coordinate
Texture X	Current X texture coordinate
Texture Y	Current Y texture coordinate
X Increment	How much to increment the X texture coordinate by
Y Increment	How much to increment the Y texture coordinate by

Table 3.5 consists of the specific variables used within the algorithm. As previously stated the minimum and maximum texture coordinates will be -1 and 1. The increment is derived from these maximum and minimum values. Equations 3.1 and 3.2 are the formulas for generating the increments:

$$xIncrement = (MaximumX - MinimumX)/ImageWidth \quad (3.1)$$

$$yIncrement = (MaximumY - MinimumY)/ImageHeight \quad (3.2)$$

The image width and height pertain to the output image. In this case, all output images will be of size 128x128 which is the same as the input or target images.

```

[1] Set current X and Y texture coordinate to the minimums;
for  $y < Image\ Height$  do
    [1] Reset texture coordinate X to the minimum;
    for  $x < Image\ Width$  do
        [1] Store texture coordinates to be used in tree evaluation;
        [2] Evaluate trees for RGB values [3] Fit the value obtained into a
            [0..255] range;
        [4] Translate the value(s) into a single RGB value;
        [5] Store the RGB value at coordinate (x, y) for the resulting image;
        [6] Increment texture coordinate X;
    end
    [2] Increment texture coordinate Y;
end

```

Algorithm 3: Procedural Texture Algorithm

The implementation of the procedural texture generation language is from [18]. This language is a very simple but efficient language. There are other languages used like fractals and perlin noise, however, those are not used for efficiency reasons. The first steps in algorithm 3 are setting up the coordinate system to traverse. The trees will generate colours through evaluating the tree expression at a specific texture coordinate. The translation of the values into a single RGB value is specific to Java's BufferedImage function setRGB() which takes the x and y coordinates followed by this RGB value. There are two options for trees: there only being one tree to generate a grayscale value; or three trees respectively generate the red, green, and blue values. The usage will be dependent on the fitness function that will be evaluating the individuals as some require colour processing, and others do not.

3.5 Direct Match

Direct match is a strict and simple algorithm used to compare two images[57]. Direct match compares the RGB channels at every pixel coordinate of the images. The summed difference of the RGB distance values will result in the fitness for the solution.

```

TotalDistance = Total distance between the RGB values ;
for Every pixel coordinate do
    [1] TotalDistance += | (Target Red - Solution Red) + (Target Green -
        Solution Green) + (Target Blue - Solution Blue) | / 3 ;
end

```

Algorithm 4: Direct Match Algorithm

3.6 Wavelet Analysis

Wavelet analysis produces wavelets derived from sound or image decomposition [52]. These wavelets describe a particular region of the sound or image. The most popular usage of analyzing wavelets is in compression. In this research, wavelet analysis will be applied to images. Wavelet analysis involves the process of continually taking away the finest details identified until a very general concept of the image remains. This process is called a wavelet transform.

3.6.1 Haar Wavelets

Haar wavelets are the simplest and are foundational to subsequent wavelet implementations [52]. The transformation operation for Haar wavelets, Haar transform, serves as the prototype for all other wavelet transforms. The Haar transform produces two subsignals which are half of the length of the signal that is provided. A running average, or trend; and a running difference, or fluctuation. Each time the

Haar transform is performed, it is known as a level. A first level Haar transform will only produce the trend and difference. However, subsequent levels use the previous trend as the discrete signal. For more information regarding the calculation of the subsignals, see [52].

A key property of Haar transforms is its ability to conserve the energy of signals, or the sum of squares of its values. Haar transforms redistribute energy in a signal through compression into a trend subsignal. The trend subsignal generated is half the length of the original signal thus being called a compaction of energy.

```

sample = Array of pixel gray values;
coefficients = Array of wavelet coefficients;
length = Length of sample;
sample = sample/ $\sqrt{Length}$ ;
while  $Length > 1$  do
    [1] Length = Length/2;
    for  $index < Length$  do
        [2] valueOne = sample[2 * index] ;
        [3] valueTwo = sample[2 * index + 1] ;
        [4] coefficients[index] = (valueOne + valueTwo)/ $\sqrt{2}$ ;
        [5] coefficients[index] = (valueOne - valueTwo)/ $\sqrt{2}$ ;
    end
    [4] sample = coefficients;
end

```

Algorithm 5: Haar Wavelet Decomposition

Algorithm 5 is the wavelet transformation for a 1 dimensional matrix. If the input data for an image is a 2 dimensional matrix instead of a transposed 1 dimensional matrix then this operation would have to be done over every row and column of the 2 dimensional matrix.

Each transformation will turn half the data into the subsignal values. After a complete Haar transformation there will be a single matrix with two pieces of data. The very first index of the matrix will be the average of all the values. In the case of an image, it will be the average value of the colour channels. All other indices within the matrix will be the coefficients of the image.

3.6.2 Fitness Function

Wavelet analysis is a method for fitness calculation of shape matching between images. A set of coefficients which determine each pixel's relative importance and average colour of the entire image are generated through this wavelet decomposition. The coefficients would help identify the structure of an image and should perform much better than that of directly matching a pixel's average value. An example of this is Gentropy [57]. Gentropy has proven that wavelet analysis works well with procedural texture evolution.

The implementation of wavelet decompositions is from [57]. The decomposition results in a 2d matrix of coefficients along with a singular value which denotes the average colour of the entire image. Many of these coefficients may be deemed as too detailed and not necessary towards the general structure of the image. With this comes the idea of truncation. Jacobs et. al [24] utilized coefficient truncation and wavelets to speed up the process of image querying in databases to great success. These 40-60 coefficients would be the largest valued, positive and negative, within the entire image. The number of coefficients they truncated to depended on the types of images which ranged from 40-60 coefficients for thumbnail images of 50x50 pixels. Using a similar ratio of 10%, the top 1600 coefficients will be looked at for 128x128 sized images.

```

CoefficientComparisonValue = Total difference in coefficient ranks ;
SolutionAverageColour = Average colour value of the entire solution image ;
TargetAverageColour = Average colour value of the entire target image ;
WaveletFitness = Final fitness value for 1600 largest coefficients do
    | if Solution coefficient position exists in Target's coefficient set then
    | | [1] CoefficientComparisonValue += | Target coefficient rank - Solution
    | | coefficient rank |
    | end
    | else
    | | [2] CoefficientComparisonValue += 2
    | end
end
[3] WaveletFitness = | TargetAverageColour - SolutionAverageColour | +
CoefficientComparisonValue

```

Algorithm 6: Haar Wavelet Truncated Fitness Function

Algorithm 6 loops through the 1600 largest coefficients while checking if the positions of the coefficients in the solution exist within the target. If there exist a coefficient in both the target and solution then an absolute difference of their ranks is computed. The coefficient rank values range from 1/1600 to 1600/1600. The absolute difference of the two ranks are taken and added onto a comparison variable. If the solution has a coefficient in a position where the target doesn't then a penalty of 2 fitness points is added to the comparison value. The final fitness formula is derived from [24].

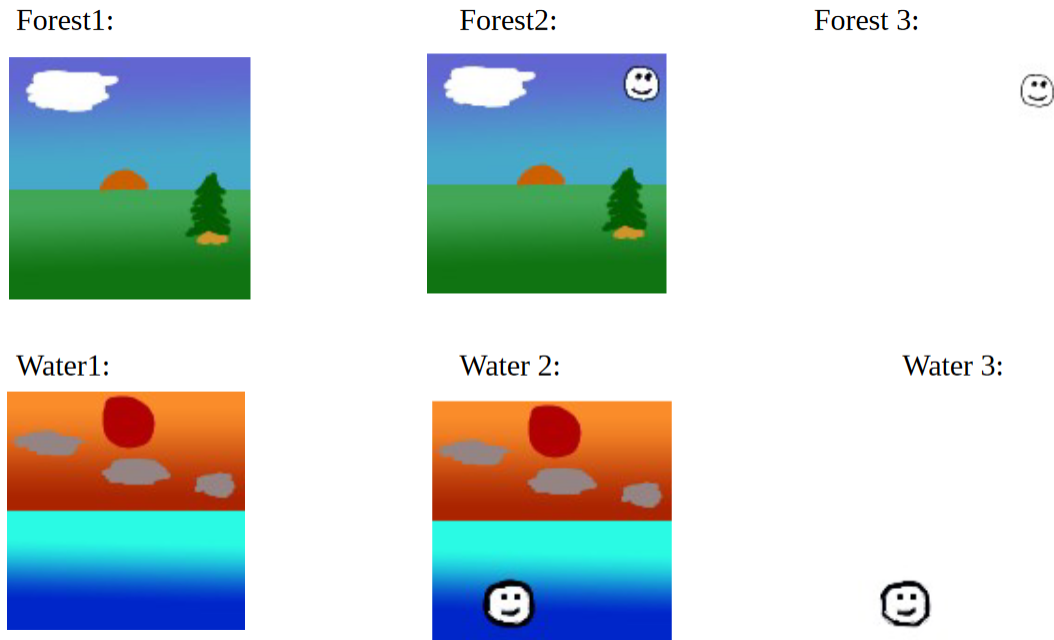


Figure 3.2: Test Images

Table 3.6: Wavelet Value Comparisons

Target Image	Comparison Image	Value
Forest 1	Forest 1	0.0
Forest 1	Forest 2	770.0000167616018
Forest 1	Forest 3	2268.0020492060603
Forest 2	Forest 3	1752.0020324444588
Water 1	Water 1	0.0
Water 1	Water 2	1158.0000383814959
Water 1	Water 3	2486.0020826205996
Water 2	Water 3	1550.0020442391035

Figure 3.2 are mock images to test the Haar wavelet functions to ensure that they work correctly. When comparing the image against itself, the value should be 0 because the highest coefficients and the average colour of the image will be the same.

When adding a slight change to the image, a smiling face in images forest 2 and water 2, the value produced should be slightly worse. This is seen in Table 3.6 when comparing forest 2 to forest 1 and water 2 to water 1. When comparing to the third image, both comparisons should be worse than that of 1 and 2 but the comparison between 2 and 3 should be better than 1 and 3. The face in the third image does not exist within the first, and the third image is only the face which is what gives the expectation for these values. This is observed in the table as well confirming that the fitness function works as expected.

3.7 SSIM

Structural Similarity (SSIM) is a popular measure that utilizes pattern comparison of pixel intensities normalized for luminance and contrast[53]. It is a way to measure image quality that focuses on structural information. This framework works off of the assumption that the human’s visual system is highly adapted to extract structural information. Even if an image is slightly blurry, ruined, or noisy, it is still possible to understand what the object or essence of the image is. This new perspective aims to evaluate the structural changes of two signals rather than directly compare things like pixel data.

There are three aspects of an image that are utilized for this assessment: luminance, contrast, and structural information. Luminance refers to the brightness or intensity of the signal, for example, a high luminance image would be extremely bright close to white and low luminance would be very dark. Contrast is the difference in luminance or colour. A high contrast image would have a lot of depth to the image where the colours, or greys, are distinct, and on the other hand, a low contrast image would be quite uniform, almost the same colour. Structural information is defined as attributes which represent the structure of objects in the scene independent of

average luminance and contrast. The local luminance and contrast of a subsection of the image is used because it can change vastly throughout the image.

They created three formulae utilizing two signals for the luminance, contrast, and structural information. These two signals would be manipulated and compared to one another, when combined, would produce the SSIM value which ranges between -1 and 1. It was also found that this value could be normalized into a range of 0 to $\sqrt{2}$ for optimization problems.

SSIM is an alternative to wavelet analysis, as it also observing the structure(shape) within images. The process of SSIM produces a single value for every window calculated. The cumulation of these values will provide a total error for the totality of the windows. There exists a more complex form of SSIM known as Multiscale SSIM or MS-SSIM. This has been proven to be used within GP. Bakurov et. al [4] utilizes MS-SSIM as a baseline to prove that their system outperforms this popular image quality assessment measure. However, for simplicity sake, only Single Scale SSIM is utilized. Within SSIM there was an addition made which tailored it towards optimization problems through the inherent properties of SSIM [6].

A sliding window strategy is proposed in [53]. 8x8 sized windows will be utilized across the entirety of the image with no intersection. This requires target images to be divisible by 8 or to have a windows that are not of size 8x8. Each normalized window value will be summed together before being averaged. This average will be the fitness value that determines the performance of a solution.

Table 3.7: SSIM Variables

Variable	Description
Window Size	The width and length of the square window
Windows	The collection of all windows
ssimValue	The running value of the SSIM windows

Table 3.7 refers to the variables necessary to perform SSIM. The window size denotes the width and lengths of the windows which have been previously identified to be 8. All the windows need to be stored or preprocessed in order to perform actions on them later. Each window will hold the pixel values and the mean grey value. The running value of SSIM will be the fitness score for the solution which needs to add all the normalized window SSIM results.

```

while Window Size * Index < Total Pixels do
    | [1] Store the window data;
end

for window in Windows do
    | [2] Calculate SSIM for the window;
    | [3] Normalize SSIM value;
    | [4] Add normalized SSIM value to ssimValue;
end

[5] Divide ssimValue by the total number of windows;

```

Algorithm 7: SSIM Fitness Calculation

The implementation of the SSIM calculations is from [47]. A window is a small image sliced out of the original target image. Following Wang et. al [53], the sliding window method will be utilized as previously mentioned. Each window will contain the average grey value and the grey pixel values. Each of these windows will then be used to calculate a SSIM value. Afterwards the normalized SSIM value will be added to the running SSIM total. The average SSIM value over the total number of windows will then be used as the comparison to the target image.

Table 3.8: SSIM Value Comparisons

Target Image	Comparison Image	Value
Forest 1	Forest 1	0.0
Forest 1	Forest 2	0.04070310959592466
Forest 1	Forest 3	0.5788680278255285
Forest 2	Forest 3	0.5620242315977924
Water 1	Water 1	0.0
Water 1	Water 2	0.05789772653466361
Water 1	Water 3	0.6423053112590693
Water 2	Water 3	0.6163984995047127

Table 3.8 utilizes the image set from Figure 3.2 to test the SSIM calculations. The pattern in the comparisons are expected to be the same as that within Table 3.6. The difference between the two is the way they go about structure comparisons. The value from SSIM is the overall difference of the windows within the image whereas haar wavelets truncate the most important pixels. The expected patterns in the comparison come out within the tests which confirm that this SSIM implementation is valid for use.

3.8 CHISTQ

Colour histograms discretize image colours through counting the occurrences of the quantized discrete colours within the image [50]. The colours are quantized into a fixed number of colours which act as histogram bins in order to count the frequency for each colour. This technique is known for image comparisons in image querying databases when dealing with colour. Its flexibility due to the binning is superior to the strictness that direct pixel matching, and has found use in GP-based texture

synthesis[57, 18].

A typical matching of colour histograms would take the difference between each bins occurrence which would be the distance for that respective bin. The total distance would be identified by summing each bins distance. Colour histogram quadratic matching, or CHISTQ, utilizes quadratic distance on top of the aforementioned colour histogram [50].

$$\begin{aligned}
 d(i, j) = & |targetHistogram_i - solutionHistogram_i| \\
 & \cdot colourSimilarity(i, j) \\
 & \cdot |targetHistogram_j - solutionHistogram_j|
 \end{aligned} \tag{3.3}$$

Equation 3.3 shows the distance at indices i and j for CHISTQ matching. The colour similarity refers to the similarity between the two quantized colours through indices i and j. The product of this with the histogram distances at each index completes the quadratic distance. This is an expensive computation since all combinations of i and j are considered, although the flexibility through colour similarity is considered enough of an advantage to outweigh the computational complexity.

CHISTQ is a fitness function for colour matching of images that works with the GP language[18, 57]. CHISTQ utilizes a distance formula which will provide the fitness. The target image and solution images are quantized into 512 discrete colour bins which will be used when calculating the quadratic distance between the two. After quantization, the two bin datasets will be used against each other in this quadratic distance formula, equation 3.3. The formula consists of a similarity calculation which is what makes CHISTQ more flexible than the direct bin difference. This formula is used for every bin and the summation is the final output of CHISTQ.

Table 3.9: CHISTQ Variables

Variable	Description
Bins	Colour frequency
Target Histogram	Histogram of the target image
Colour Distance	CHISTQ output value

The variables in Table 3.9 are necessary for the process of performing CHISTQ. The bins are stored in a 1-d matrix of the discrete colours which hold the frequency of each colour from an image. The target histogram is the bins or histogram generated from the target image. Colour distance is the output of the summation of all the quadratic distances.

```

for pixel in Image do
    | [1] Increment according colour bin based off of pixel value;
end

for index in Bins length do
    | for index2 in Bins length do
    | | [2] Add quadratic distance of the histogram indices to colourDistance;
    | end
end

```

Algorithm 8: CHISTQ fitness calculation

The implementation of CHISQ is from [18]. During quantization, a colour value is derived from every pixel and the respective bin is incremented to calculate the frequency in order to generate the histogram of an image. After the histogram is generated for the target and the comparison the distance formula is used. As previously mentioned, equation 3.3 is performed on every bin. Each bin in the comparison is checked against every bin in the target for the similarity. This makes the call expensive, however, the flexibility of the algorithm is gained.

Table 3.10: CHISTQ Value Comparisons

Target Image	Comparison Image	Value
Forest 1	Forest 1	0.0
Forest 1	Forest 2	7.542183739613777E-4
Forest 1	Forest 3	1.838177538034131
Forest 2	Forest 3	1.7880744659557657
Water 1	Water 1	0.0
Water 1	Water 2	0.002575007836032119
Water 1	Water 3	1.7887742074748882
Water 2	Water 3	1.7032111615537806

As with SSIM and Haar wavelets, the patterns of images 1 and 2 being smaller, 1 and 3 being larger, and 2 and 3 being slightly less than the comparison between 1 and 3 remains the same. With the focus this time being on the colours and the similarity in a colour respective to the whole image the patterns should still emerge. In figure 3.2, the colours between the first two images of forest and water should be vastly more similar than with the third image. Whereas the face provides some semblance of similarity when compared to the second image.

Chapter 4

Procedural Texture Experiments

4.1 Direct Match Experiment

This experiment will be utilizing a direct match fitness function in accordance with behaviours relating to the colour channels. The experiment will be comparing vanilla GP, island model distributed evolution, high-dimensional MAP-Elites, and distributed MAP-Elites. A target image will be provided to each of these tests where they will produce procedural textures. In the case of this experiment, all tests will be utilizing the target image will be the image in Figure 4.1.



Figure 4.1: Target Image








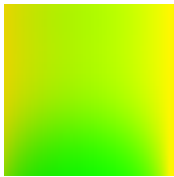
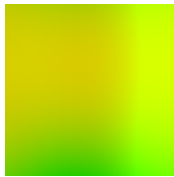
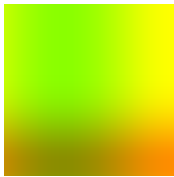
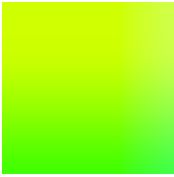
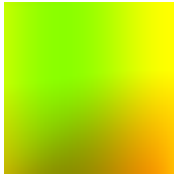





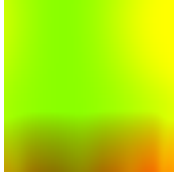
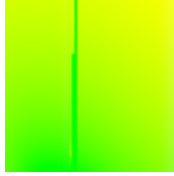

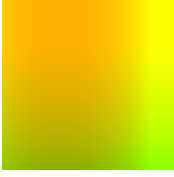






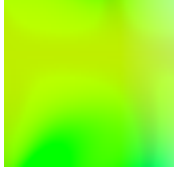

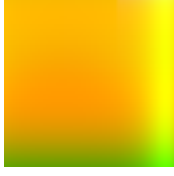
Fitness Function	Direct Match Luminosity/ Grayscale	RGB distance: channel distance between Target and GP Image (pixels).
Behaviour 1	Average Red Channel	Averaged the green channel over all the pixels.
Behaviour 2	Average Green Channel	Averaged the green channel over all the pixels.
Behaviour 3	Average Blue Channel	Averaged the blue channel over all the pixels.

Table 4.1: Fitness Function and Behaviours

Direct Match fitness as described in Table 4.1 is a very strict and basic fitness function which calculates the error between each pixel's red, green, and blue distance between the target and GP image. This direct match fitness will be used in all islands and sub-maps for the distributed algorithms. Although the distributions are capable of housing different fitness functions, they will share the same fitness here.

4.1.1 Vanilla GP Results

Table 4.2: GP Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

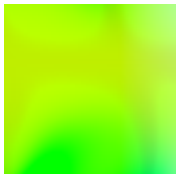
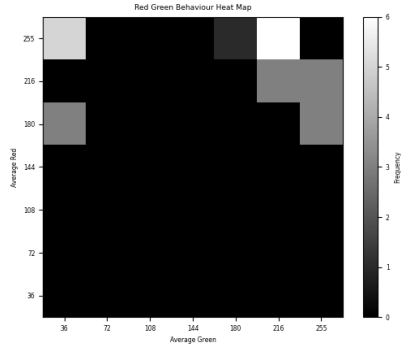
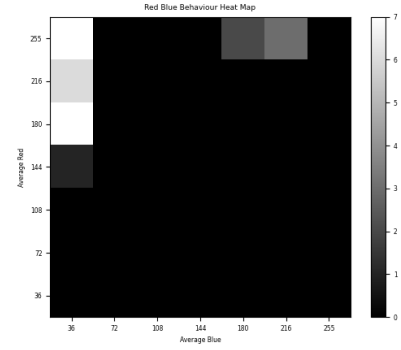


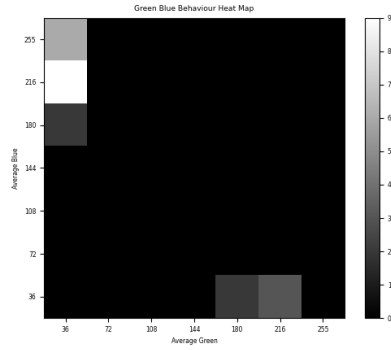
Figure 4.2: Best Solution



(a) Red and Green Channel Heat Map



(b) Red and Blue Channel Heat Map



(c) Green and Blue Channel Heat Map

Figure 4.3: Colour Channel Heat Maps

The best solutions evolved from 30 runs show similar tendencies to one another with slight variances. The runs generally chose a single gradient between a few colours and have a darker shaded design on top of it. For example some of the purple-blue solutions and orange-green have a pattern of some kind of hill-like structure in the bottom left. There are multiple designs generated, however, none resemble anything close to the target image.

Figure 4.2, the best solution out of the 30 runs, has the most interesting pattern out of all the runs. It chose a more green palate and has something resembling an H-like shape that is darker than the rest of the image. When comparing to the Van Gogh target image, there is nothing similar about it. Looking at it from a different

angle, the behaviours of the mean red, green, and blue channels will be compared. For reference, the Van Gogh image resides in the behaviour intervals for red 108-144, a green of 144-180, and blue of 108-144. This solution has intervals of red 144-180, green 216-255, and blue 0-36. The only behaviour close to the target image in this case is the red channel while green is slightly more distant and blue is a far cry away.

Figures 4.3b, 4.3a, 4.3c which represent the diversity of solutions throughout the 30 runs of this experiment. The heat map axes are the same behaviours used within the ME and DME experiments, however, formatted in the DME representation. This formatting is to represent the correlation between each colour channel. The Red-Blue and Green-Blue heat maps have the most overlap in solutions in comparison to the Red-Green indicating that solutions have the most diversity between red and green. Regardless of the red and green values, the blue channel does not change much.

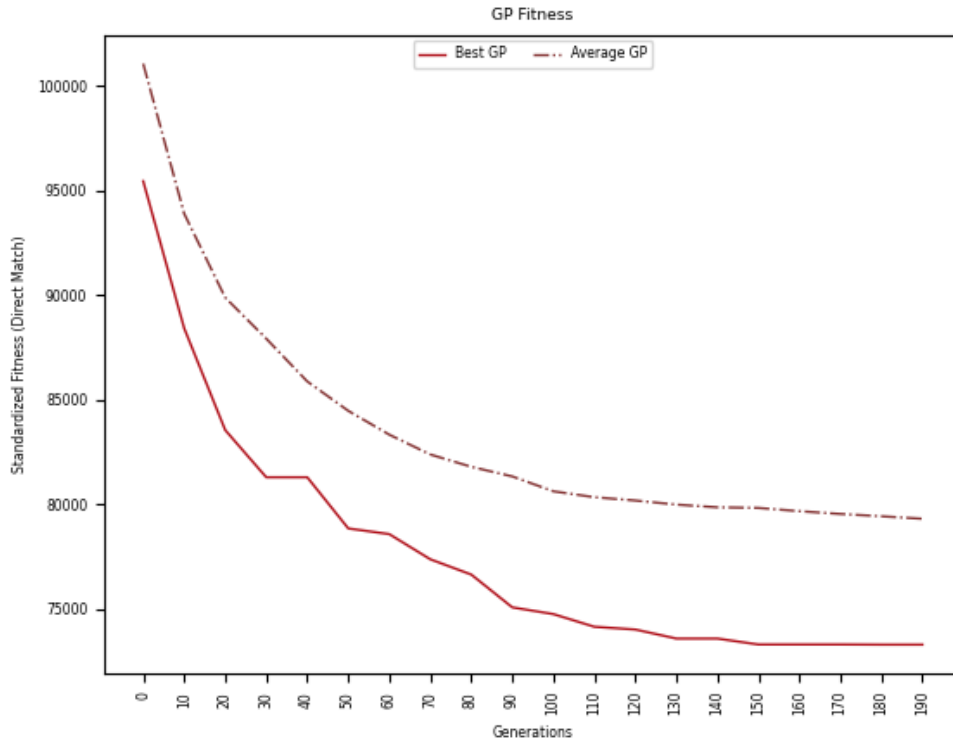
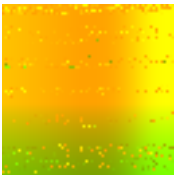

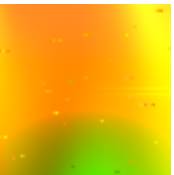





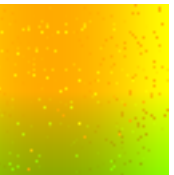
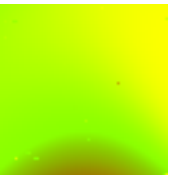
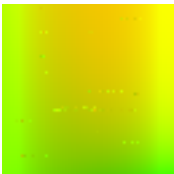

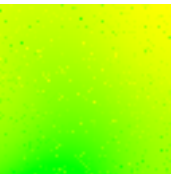



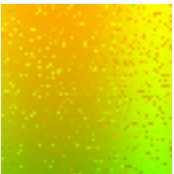




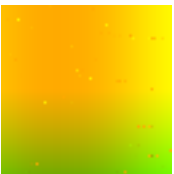




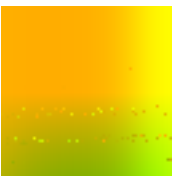





Figure 4.4: Best GP Fitness Graph

The plot in Figure 4.4 shows the best and average fitness of the best solution over 30 runs. The average fitness values improve greatly over the first 20 generations before slowly improving over the next 80 generations. After generation 100 it begins to very gradually improve before seemingly begin to plateau on the last generations. The best fitness at each generation makes the most improvement within the first 42 generations before slowly improving. After generation 90 the plateaus are more noticeably longer.

4.1.2 Island-Model Results

Table 4.3: IM Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

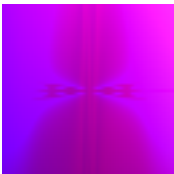
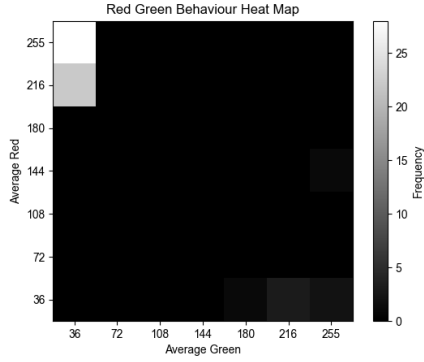
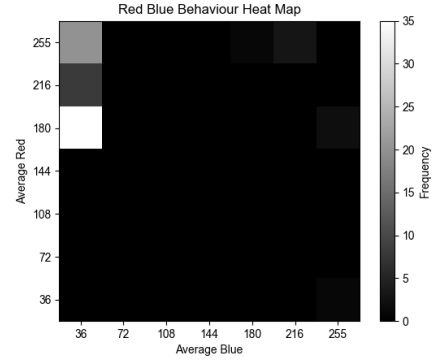


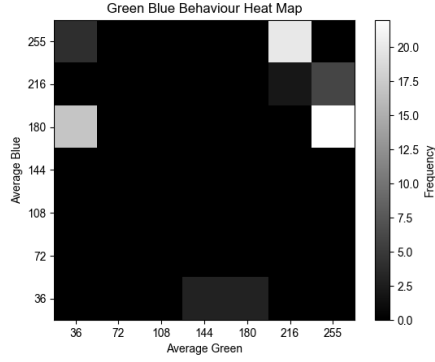
Figure 4.5: Best Solution



(a) Red and Green Channel Heat Map



(b) Red and Blue Channel Heat Map



(c) Green and Blue Channel Heat Map

Figure 4.6: Island-Model Colour Channel Heat Maps

Table 4.3 show the best solutions for each run. Since there are three subpopulations the image shown is one with the best fitness out of their respective subpopulations. The best solutions generated generally follow the same pattern as the results found within the vanilla GP experiment. There are some interesting differences like the seemingly random pixels across some of the runs as well as some differing patterns.

The best solution in Figure 4.5 is the most visibly interesting in comparison to all the others. The deeper purple has interesting geometry as well as the lighter stripes coming down at a diagonal from the middle. This image comes from the second subpopulation. When comparing behaviours to the target image with behaviour intervals of red 108-144, a green of 144-180, and blue of 108-144, this solution resides

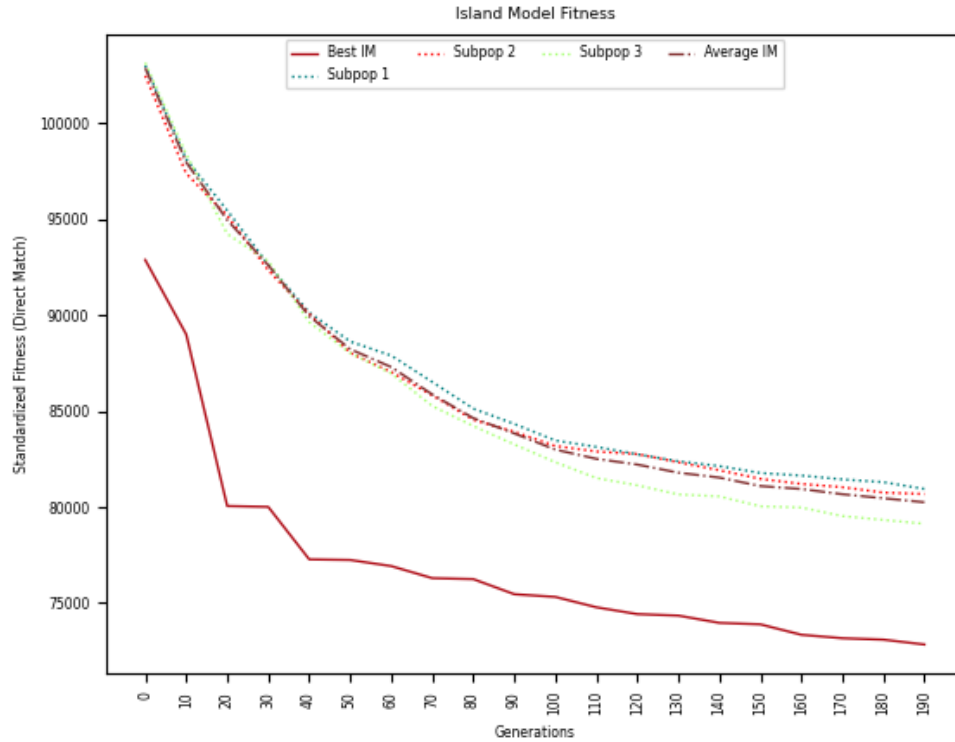


Figure 4.7: Best Fitness

in a red of 180-216, a green of 0-36, and a blue of 216-255. The average red and blue of the image are close to that of the target image where the green is very far from it.

Figure 4.6 is the heat map representation of the island-model evolution under the same fitness function of the vanilla GP experiment. The heat maps dataset here utilizes every best individual from each subpopulation resulting in 90 individuals to be evaluated. There is much more diversity between blue and green here than with red. The Red-Green and Red-Blue heat maps are very concentrated whereas Blue-Green is much more spaced. The relationship showing that there are more varied results with average blue and green values and fewer red.

The fitness graph in Figure 4.7 shows a relatively smooth evolution until a convergence around generation 190 for the average fitness of the subpopulations as well as all the generations. The jaggedness of each subpopulation is the result of immigration

and emigration of individuals where the best individual of a subpopulation would be emigrated out. Depending on the next best individual the increase in fitness may be quite sharp if a individual of similar fitness to the previous best is not immigrated in. The best fitness values also evolve the most in the first 50 generations like the average.

4.1.3 MAP-Elites Results

Table 4.4: Behaviour Intervals

Behaviour Name	Total Intervals	#1	#2	#3	#4	#5	#6	#7
Mean Red	7	0	36	72	108	144	180	216
		35.9	71.9	107.9	143.9	179.9	215.9	255
Mean Green	7	0	36	72	108	144	180	216
		35.9	71.9	107.9	143.9	179.9	215.9	255
Mean Blue	7	0	36	72	108	144	180	216
		35.9	71.9	107.9	143.9	179.9	215.9	255

This algorithm utilizes a cube(hypercube) format where a visualization would result in a 3-dimensional cube. The behaviours which are calculated for these individuals within the algorithm are highlighted in Table 4.1. These behaviours all will have 7 intervals which will result in a maximum of 343 possible individuals within the cube. The intervals for each behaviour are in table 4.4. Within each respective interval the value higher up in the cell is the lower limit and the one at the bottom of the cell is the upper limit, for example, interval 1 would range from values 0-35.9

Table 4.5: ME Image Results



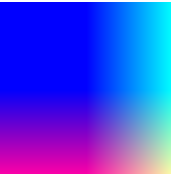

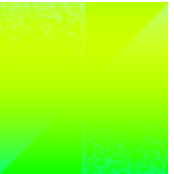

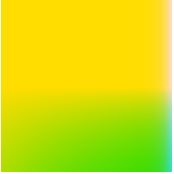

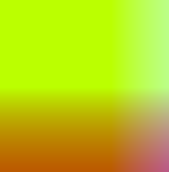


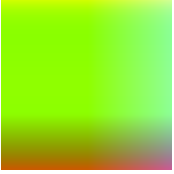



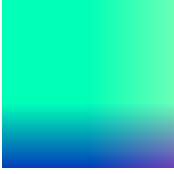











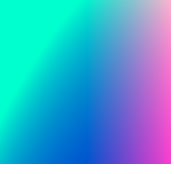


Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				



Figure 4.8: Best Solution

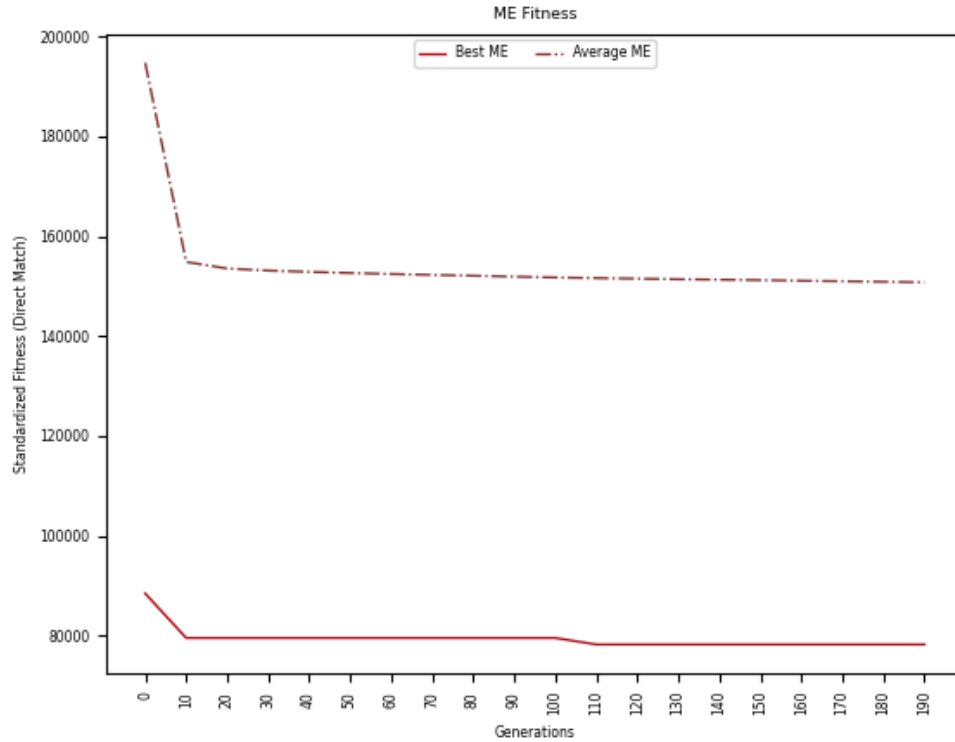


Figure 4.9: Average Fitness

Table 4.5 is the compilation of the best results from each run. The runs tend to perform a diagonal or vertical gradient. There are some exceptions, but overall the solutions do not have any semblance of shapes and mix colours together. There are some interesting ways the colours curve into one another but they do not have a cohesive shape to them.

Figure 4.8 is the best solution from the set of runs. It seems like a mostly light green image with a slight gradient at the sides. Van Gogh has behaviour intervals of red 108-144, green 144-180, and blue 108-144. This solution has a red of 144-180, a green of 216-255, and a blue of 0-36. The lack of blue is apparent in the image, however, the red is quite close to the target image.

Figure 4.9 is the fitness graph averaged over 30 runs for 200 generations. Both the average and best quickly improve within the first 10 generations, or a 22500

evaluation equivalent. The average fitness of the MAP is observed to have a much smoother descent after the rapid improvement while there are significant jumps within the best individuals observed. There is a large difference in the fitness values between the mean and best due to the nature of MAP-Elites. Previously empty cells may be filled with poor solutions which may never be replaced. If these individuals do get replaced they are not necessarily vastly better than their predecessor which results in a higher average MAP fitness. The best fitness at each generation is primarily

4.1.4 DME Results

Table 4.6: Direct Match DME Sub-MAPs

	Sub-MAP 1	Sub-MAP 2	Sub-MAP 3
Behaviour 1	Mean Red Value	Mean Red Value	Mean Green Value
Behaviour 2	Mean Green Value	Mean Blue Value	Mean Blue Value

DME utilizes multiple 2-dimensional MAPs with each behaviour pairing in order to encapsulate x dimensions. In this case there are 3 pairings of behaviours. These behaviours utilize the same interval count and intervals as those in the MAP-Elites algorithm (Table 4.4). Each sub-map will have a maximum individual count of $7^2 = 49$. This configuration of 3 sub-maps will result in a total count of $(7^2) * 3 = 147$ individuals. Table 4.6 shows the configuration of each island's MAP behaviours.

Table 4.7: DME Image Results



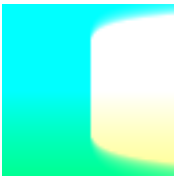
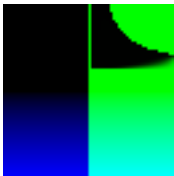


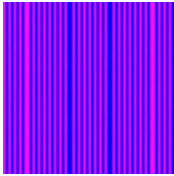
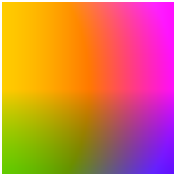


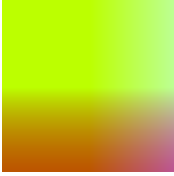



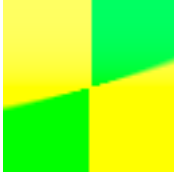




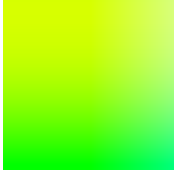



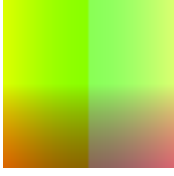
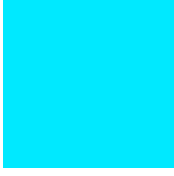
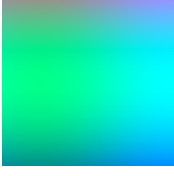


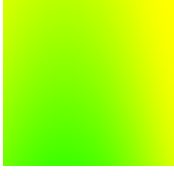
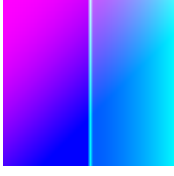
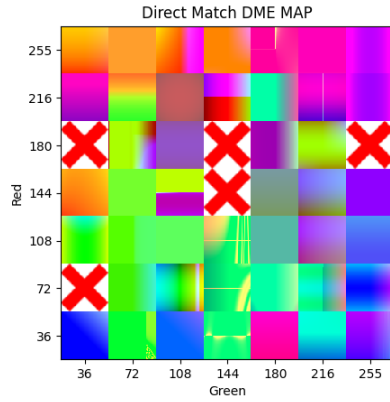
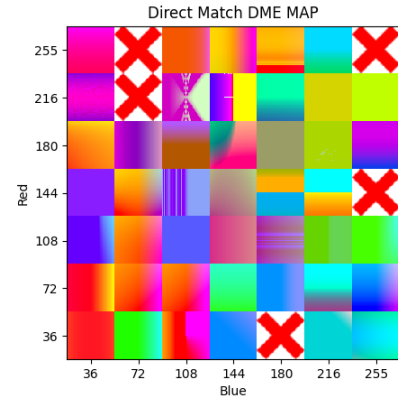
Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				



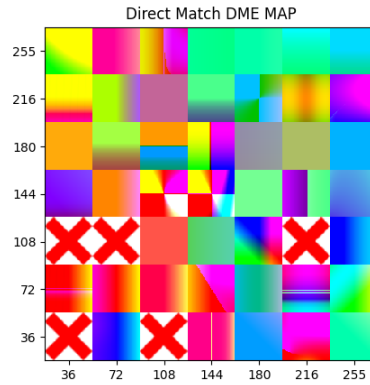
Figure 4.10: Best Solution



(a) Red and Blue Island MAP



(b) Red and Green Island MAP



(c) Green and Blue Island MAP

Figure 4.11: Direct Match DME Visualization

The best solutions produced from the 30 runs in Figure 4.7 depict a common evolutionary trait among all the runs. They all have some sort of colour gradient throughout the entire image with some of the solutions having patterns over top of these colour gradients. There are exceptions to these observations like runs 5, 8, 13, 19, and 27. These runs have specific geometry or patterns as their best solutions.

The best solution in Figure 4.10 is a gradient of blue to purple from bottom to top with some interesting spikes in the top right and bottom right protruding towards the middle. The Van Gogh target image having a red interval of 108-144, green of 144-180, and a blue of 108-144. This best solution has a red of 108-144, green of 0-36,

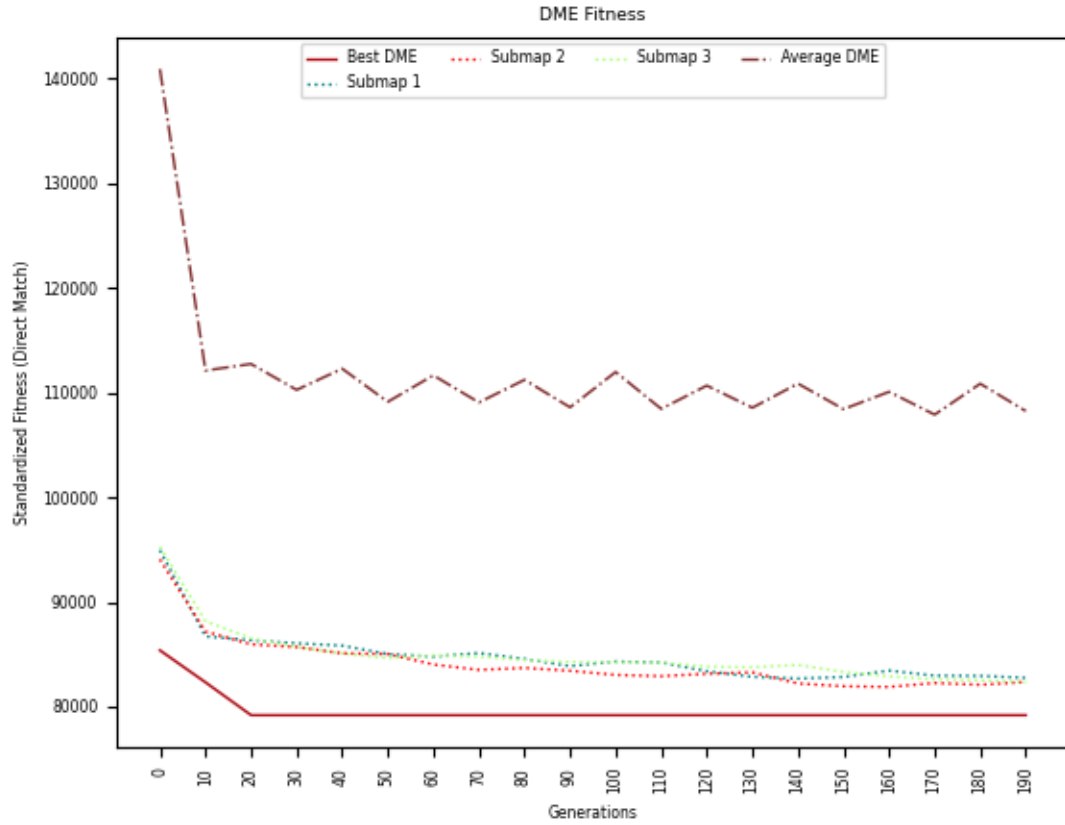


Figure 4.12: Best Fitness

and a blue of 216-255. It successfully occupies the same red interval but strays quite far from the green and blue.

Figure 4.11 shows a visualization of the MAP configuration holding the best solution of Figure 4.10. The red X's with a white background indicate that there were no solutions found with the according behaviours of that cell. There are some interesting solutions between all three MAPs with distinct geometry or patterns evolved but nothing visually similar to the target.

The average fitness in Figure 4.12 is very jagged and inconsistent due to the immigration and emigration of individuals between MAPs. Around 10% of the maximum MAP population being immigrated and emigrated frequently cause consistent spikes between the sub-maps in the average fitness graph. There are sub-maps which have

much more drastic increases and decreases due to the nature of these immigrations and emigrations since the best individual may be pulled out. As a result of this, the new island for this individual may have much more comparable fitness values and not be the best. The first and third sub-maps start at off with worse individuals than the second but quickly get competitive. All three sub-maps improve at a fast rate until generation 20 before slightly slowing down until generation 135 which they then proceed to bounce around. The best fitness sees a few points of improvement, around generation 20 and 190.

4.1.5 Discussion

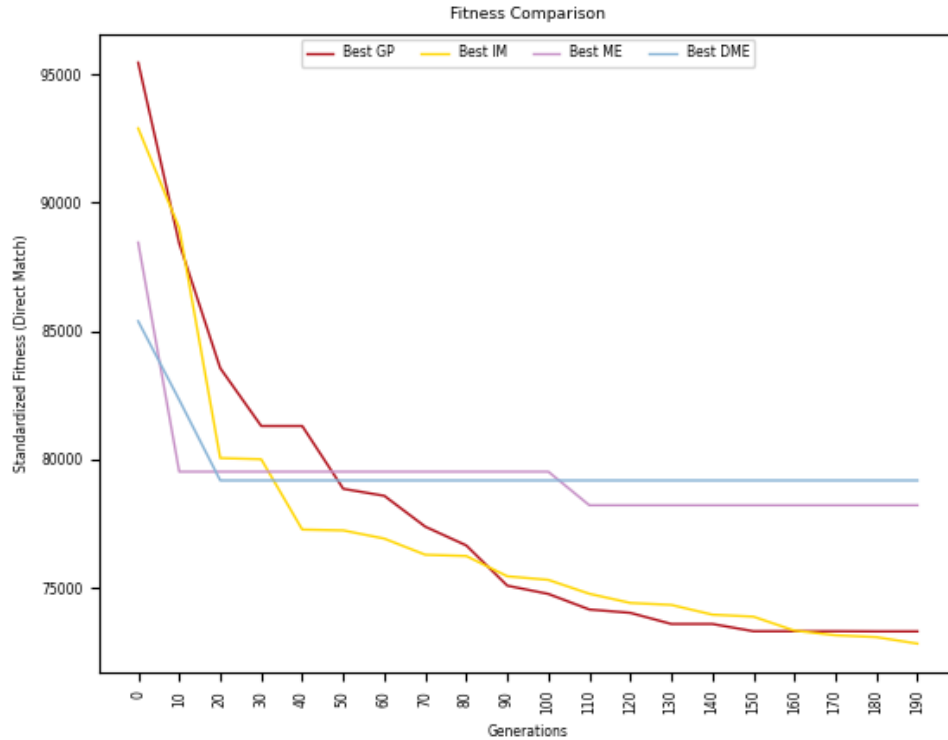


Figure 4.13: Best Fitness Comparison

Figure 4.13 is a comparison of each algorithm used within the experiment utilizing the direct match fitness function. Island model evolution and vanilla GP have similar

final fitness values and curves because of the similar evolutionary basis. Island-model introduces some variance through immigration and emigration which may attribute to the slightly worse fitness in comparison to vanilla GP. This immigration and emigration also results in a rocky fitness graph since the possible emigration of the best individuals accompanied by the proceeding reproduction in these subpopulations. Island-model's values are still competitive with vanilla GP, however, fine tuning the parameters may result in better performance due to the diversity but the fine tuning does not guarantee the super linear speed up found in other research. MAP-Elites converges very quickly and slowly improves afterwards. The maximum size of the MAP introduces space for individuals with lesser fitness values to populate. This may result in the poor initial generational fitness values, but quickly result in a convergence. The DME configuration introduces a lot less space for individuals to populate in comparison to MAP-Elites which results in a faster convergence. The early individuals also seem to outperform vanilla evolution but the convergence does not differ much from the initial fitness values. Vanilla GP by far performs the best in comparison to all others. DME performs the worst out of all of the algorithms, however, the difference in fitness values is not egregious. ME is comparable to vanilla GP and island model and could possibly perform better given more generations observing the small sharp descents over time.

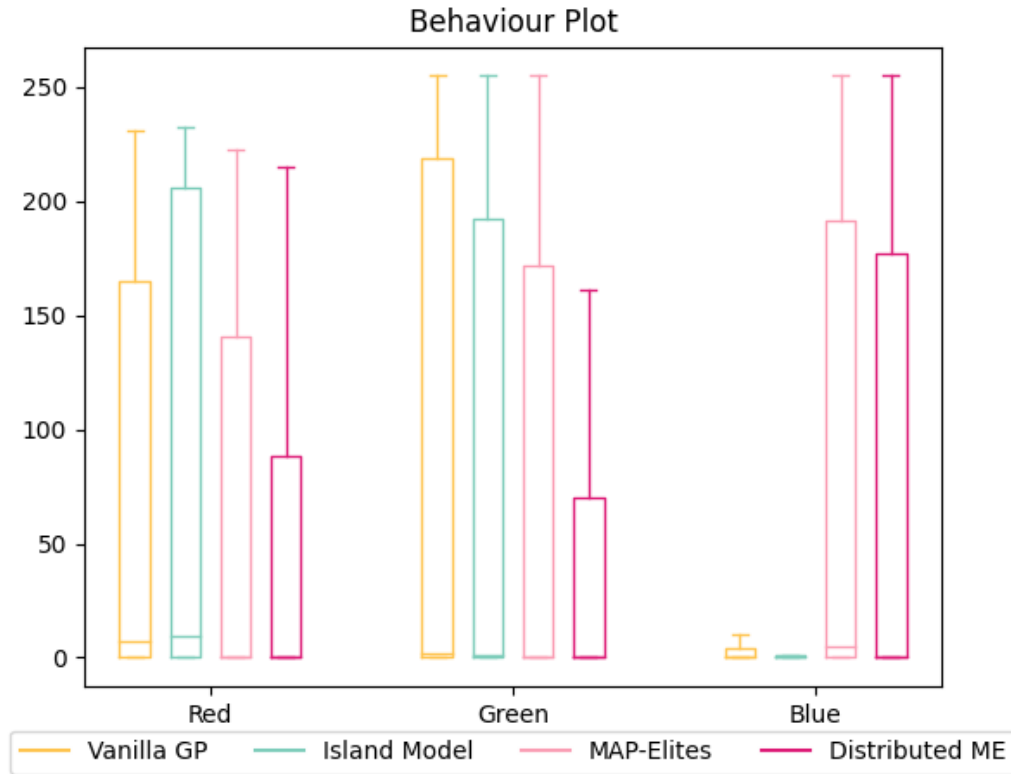


Figure 4.14: Behaviour plot using the best solutions from each algorithm

Figure 4.14 is a box and whisker plot excluding outlier points for the four algorithms used within this experiment on each of the behaviours. Outliers are defined as values beyond 1.5 times the interquartile range (25^{th} and 75^{th} percentile). The three behaviours observed are the red, green, and blue colour channels. All of the algorithms have best solutions which span the full colour range of red and green values. The blaring difference is the observed blue channel of these best solutions. DME and MAP-Elites have large interquartile boxes as well as whiskers for the blue channel showing more diversity in best solutions when compared vanilla GP and island-model evolution.

Figure 4.15 illustrates the tree depths of the best solutions obtained from the 30 runs, because of this, island-model and DME will have more data points to work with due to the nature of said algorithms. Vanilla GP and island-model produce best

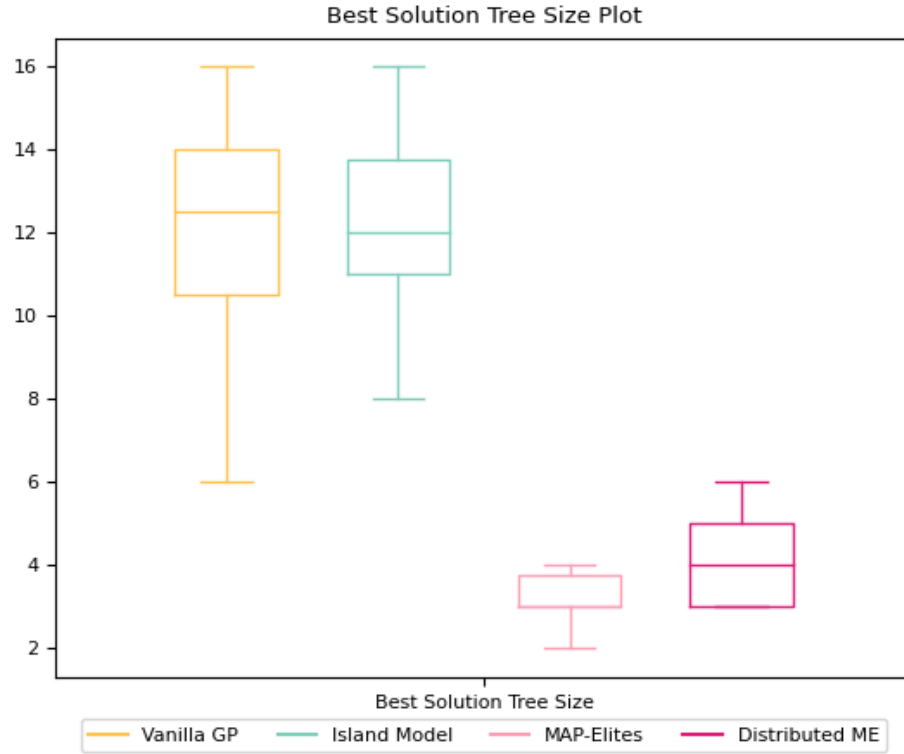


Figure 4.15: Tree Depth Box Plot

solutions with a median around depth 12 with interquartile ranges from 6-16 and 8-16 respectively. Larger trees are not inherently bad but cause longer evaluation times and possibly have sub-trees that do not do anything due to the sheer size. ME roughly produces best solutions around depth 3 and DME at depth 4 with respective interquartile ranges of 2-4 and 3-5. ME and DME produce best solutions which may perform worse than GP and island-model at a small margin but with trees that are much smaller and require much less to compute.

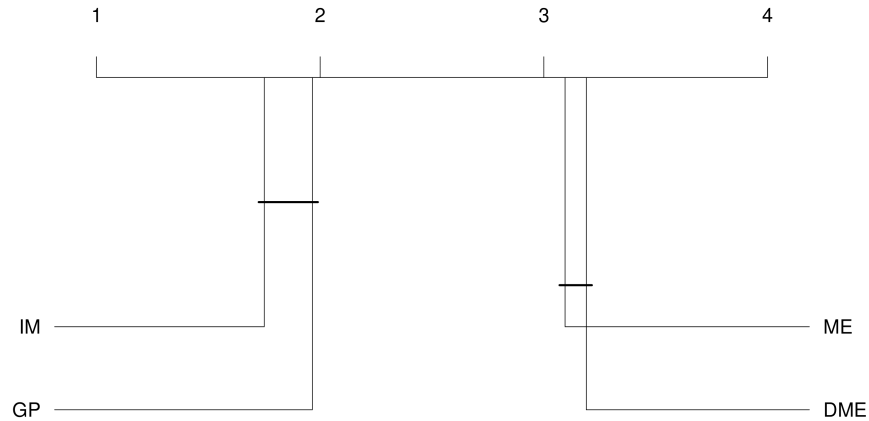


Figure 4.16: Critical Difference Diagram using best solutions from each algorithm

All following statistics have been formulated within R utilizing the Friedman test and Shaffer correction with a threshold of $p < 0.05$, with the null hypothesis:

$$H_0 : \theta_{GP} = \theta_{IM} = \theta_{ME} = \theta_{DME}$$

The Friedman test and Shaffer correction were chosen in part of the influence of [12]. The Friedman test ranks algorithms for problems separately before averaging the ranks to give a final rank to an algorithm. Figure ?? shows that IM and GP are not statistically significant to each other, and the same is said for ME and DME. However, the ME and vanilla variants are statistically significant to one another.

In summary, vanilla GP and IM performed better in terms of pure fitness, however, the 30 runs produced very similar looking solutions overall. ME produced solutions with more variance in the 30 runs but performed much worse than GP and IM. DME produced the most diverse set of individuals while performing closer in terms of fitness. The quality of all the solutions is very poor. This can be mainly attributed to the GP language which is far too simple for this target image. The fitness function being strict does not help with this either.

4.2 Wavelet Experiment

The experiment within this section will focus on exploring Haar wavelet transformations as the fitness function using a different set of behaviours. As with the Direct Match experiment, this experiment will also utilize all four algorithms. However, all of the image data will be processed as grey values. The Shaffer procedure was found more to be more beneficial in comparison to the Holm procedure. The difference is that instead of high-dimensional MAP-Elites only 2 dimensions will be configured. There will be two target images for this experiment. Figure 4.17 is intended to be a simpler target image for the fitness function to fulfill whereas Figure 4.18 is a more difficult target to solve.

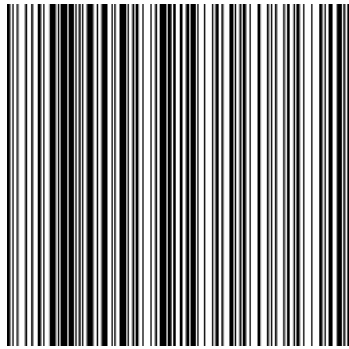


Figure 4.17: First Target



Figure 4.18: Second Target

Fitness Function	Haar Wavelet Coefficient Comparison	Ranking x highest value coefficients and comparing the respective indice's rank.
Behaviour 1	Entropy	Calculating entropy over the image's pixels.
Behaviour 2	Average Luminosity	Averaging all the colours within the image.

Table 4.8: Fitness Function and Behaviours

Table 4.8 describes a more flexible fitness function in wavelet comparisons due to wavelets identifying coefficients which show importance rather than calculating an error over every pixel as in direct match. The fitness function will remain consistent for all island configuration algorithms in this experiment as well.



Figure 4.19: Barcode Coefficients

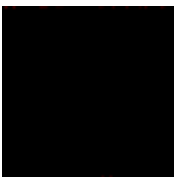
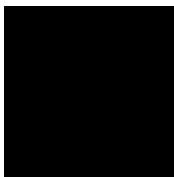
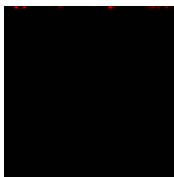
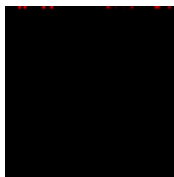

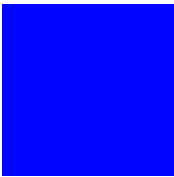

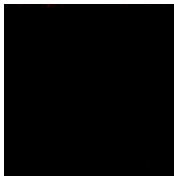
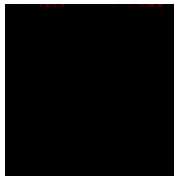





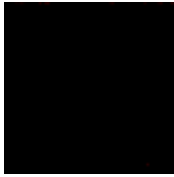
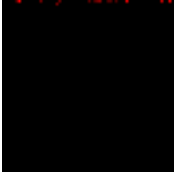

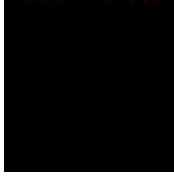
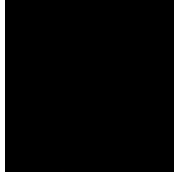


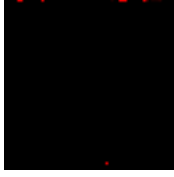
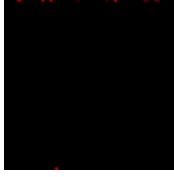

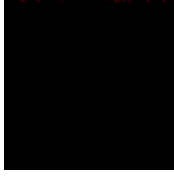
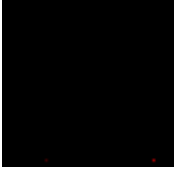
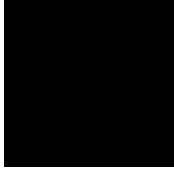
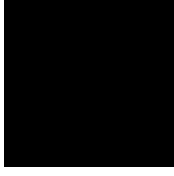




Figure 4.20: Smile Coefficients

Figures 4.19 and 4.20 are the pixels identified as having the highest value coefficients in each of the target images.

4.2.1 Vanilla GP Results: Barcode

Table 4.9: GP Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

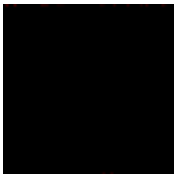


Figure 4.21: Best Solution



Figure 4.22: Best Solution Coefficients

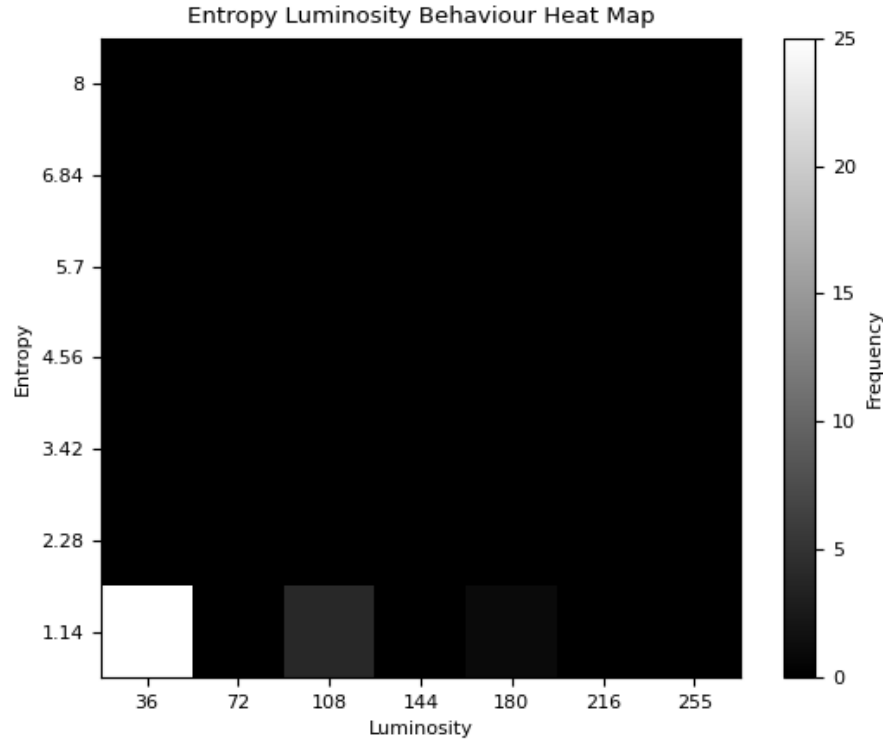


Figure 4.23: Entropy Luminosity Heat Map

The best solutions from the 30 runs in Table 4.9 show a very poor performance with the vast majority of solutions being a flat image with some sparse amounts of red littered throughout the top or bottom of the image. Some runs choose to do a blue or green variation of this.

The best solution furthers this with the behaviour evaluation used on it. It produces an entropy value that would be between 0-1.14 and a luminosity between 72-108. In comparison, the target image of a barcode has an entropy of 3.42-4.56 and a luminosity of 180-216. The coefficients which it identified as the most important show a slightly different story than the visual performance. Visually it seems to be similar to those of the coefficients identified by the target image.

The heat map, Figure 4.23, of the individual being placed into the MAP further the poor diversity performance by the vast majority of runs being in a single cell with

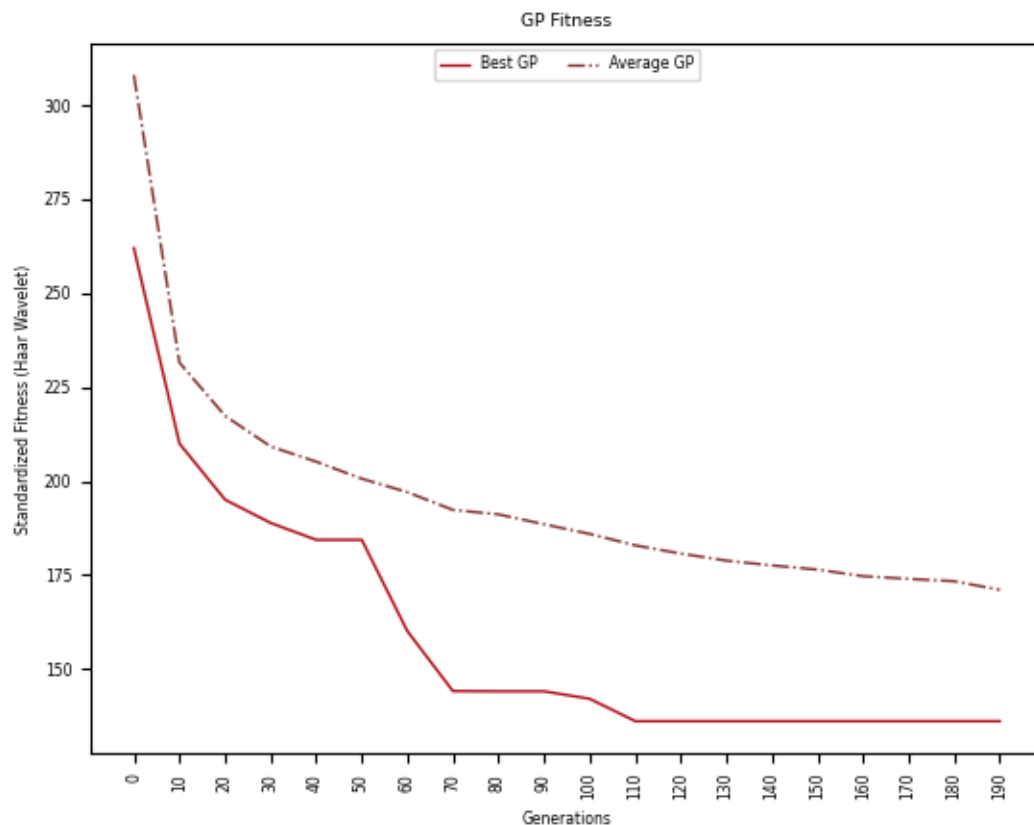


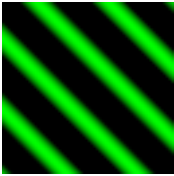
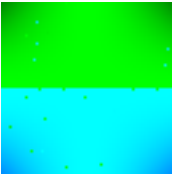
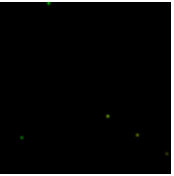


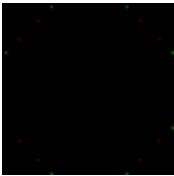
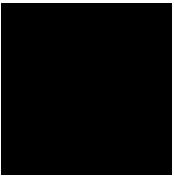
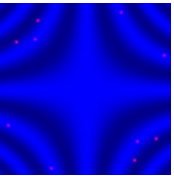
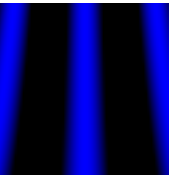
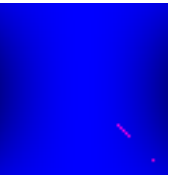
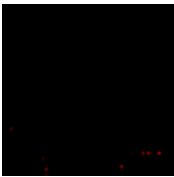
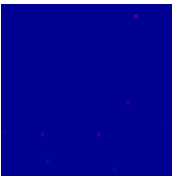
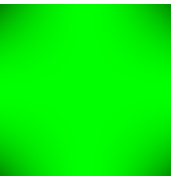
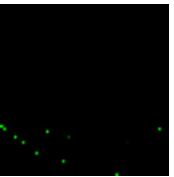
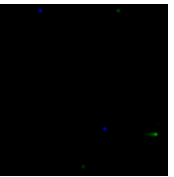
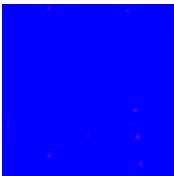
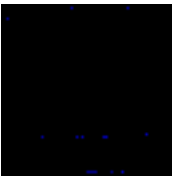
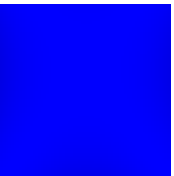

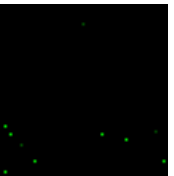
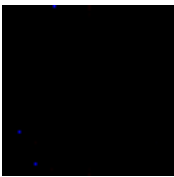
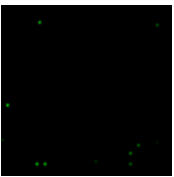
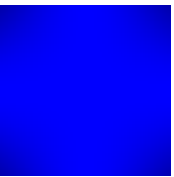

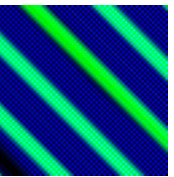
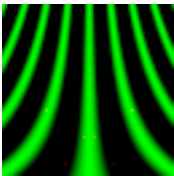
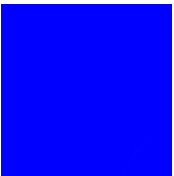
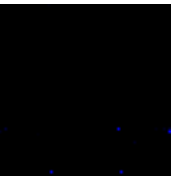

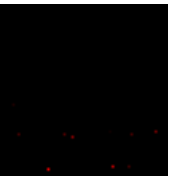
Figure 4.24: Best Fitness

the others split into two other cells.

Vanilla GP seems to evolve the quickest within the first 5 generations. It slows down into a gradual evolution of fitness until generation 110 where it slows greatly. Generation 190 is seen to be the start of the plateau of fitness. The best fitness makes bursts of improvement within the first 10 generations and between generations 50 and 70, otherwise improving slightly or plateauing.

4.2.2 Vanilla GP Results: Cartoon Smiling Face

Table 4.10: GP Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

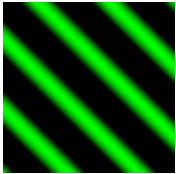


Figure 4.25: Best Solution



Figure 4.26: Best Solution Coefficients

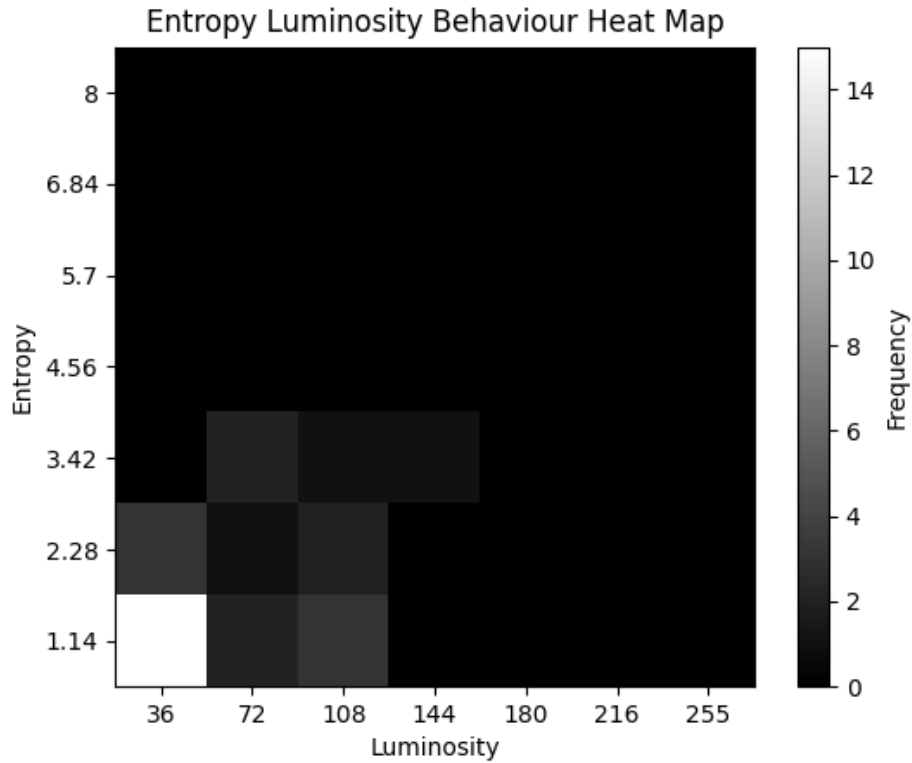


Figure 4.27: Entropy Luminosity Heat Map

Vanilla GP produces some interesting images while trying to evolve for a cartoon smiling face in Table 4.10. The best solutions from each run do not evolve well on a simple language and have many runs producing very visually poor results.

The best solution out of these are diagonal stripes in Figure 4.25. The coefficient derived from this solution also differs wildly in comparison to the target's coefficients. This best solution would fit into the cell with entropy 2.28-3.42 and luminosity of 0-36. The cartoon smiling face fits into the cell of entropy 2.28-3.42 and luminosity 180-216. It successfully matched the entropy, however, the luminosity falls very short.

In the heat map, Figure 4.27, it is observed that GP diversity falls quite short over 30 runs. There are no solutions in the heat map which occupy the same cell as the target image with the vast amount of runs producing solutions which would occupy entropy 0-1.14 and luminosity 0-36. It does successfully produce solutions with the

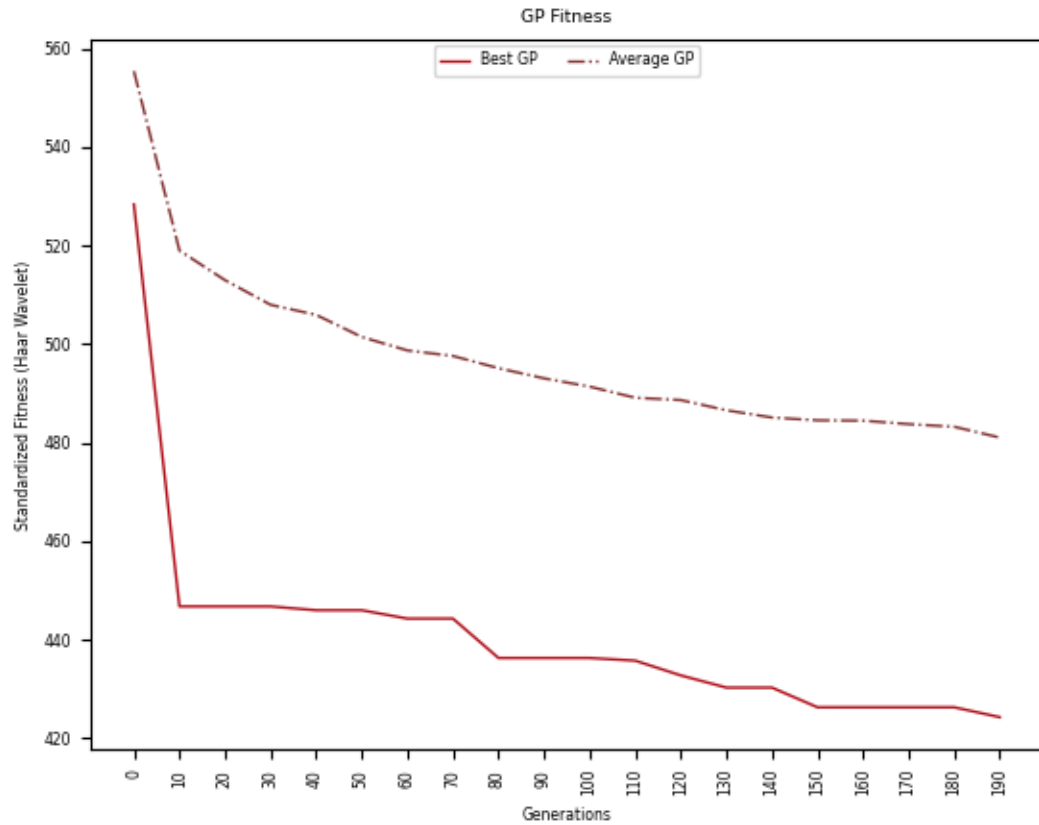


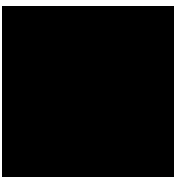


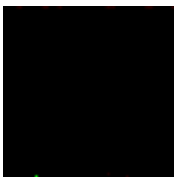

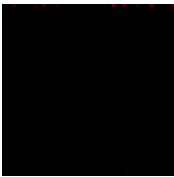
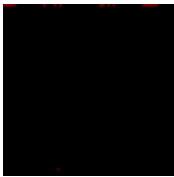

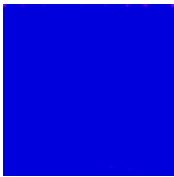



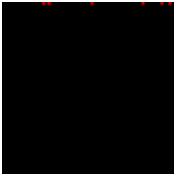

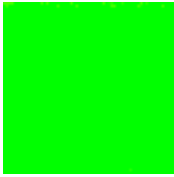

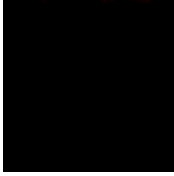
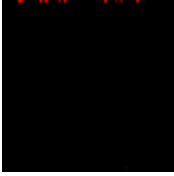
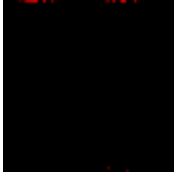
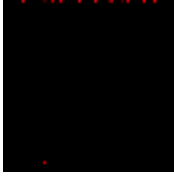
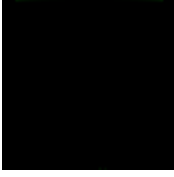

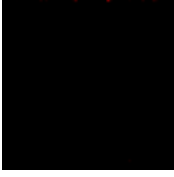
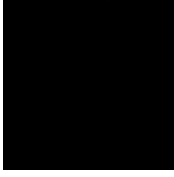
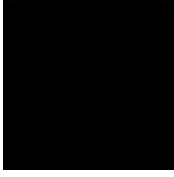
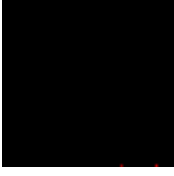


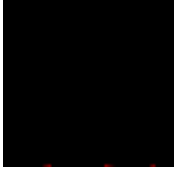

Figure 4.28: Best Fitness

same entropy with low frequency cells.

The best fitness in Figure 4.28 shows the fitness improving drastically from generation 0 to 15 before slowing down. It proceeds to slowly evolve until generation 135 where it begins to plateau. At generation 190 there is another spurt of evolution before stopping. The majority of improvement in the best fitness are within the first 20 generations, afterwards, it goes on long strides of plateaus before incremental improvements.

4.2.3 Island-Model Results: Barcode

Table 4.11: IM Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

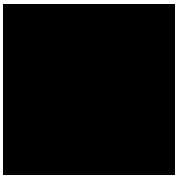


Figure 4.29: Best Solution



Figure 4.30: Best Solution Coefficients

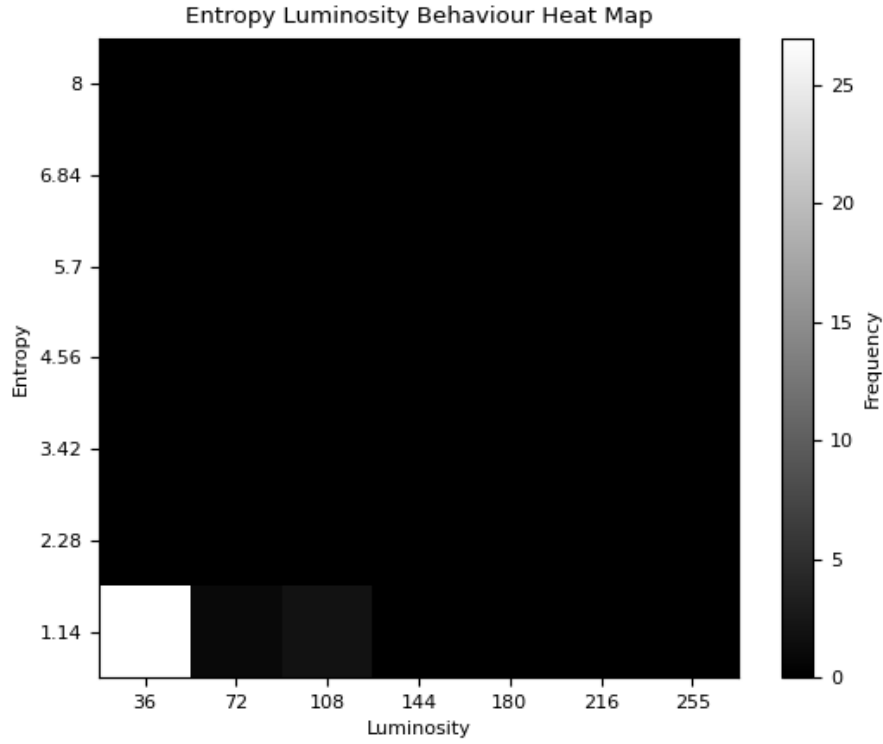


Figure 4.31: Entropy Luminosity Heat Map

Over the 30 runs using island-model evolution and Haar wavelet transformations the solutions are all visually very poor. Vast amounts of flat images either being black or blue with some solutions having sparse amounts of red on the top or bottom of the image.

Interestingly, the best solution's coefficients in Figure 4.30 is quite visually similar to the coefficients used in the target image. However, the behaviours evaluated for this best solution provide the opposite story where it would slot into a cell with an entropy of 0-1.14 and a luminosity of 0-36.

The heat map in Figure 4.31 indicates very poor diversity which corroborates the visual observation of Table 4.11.

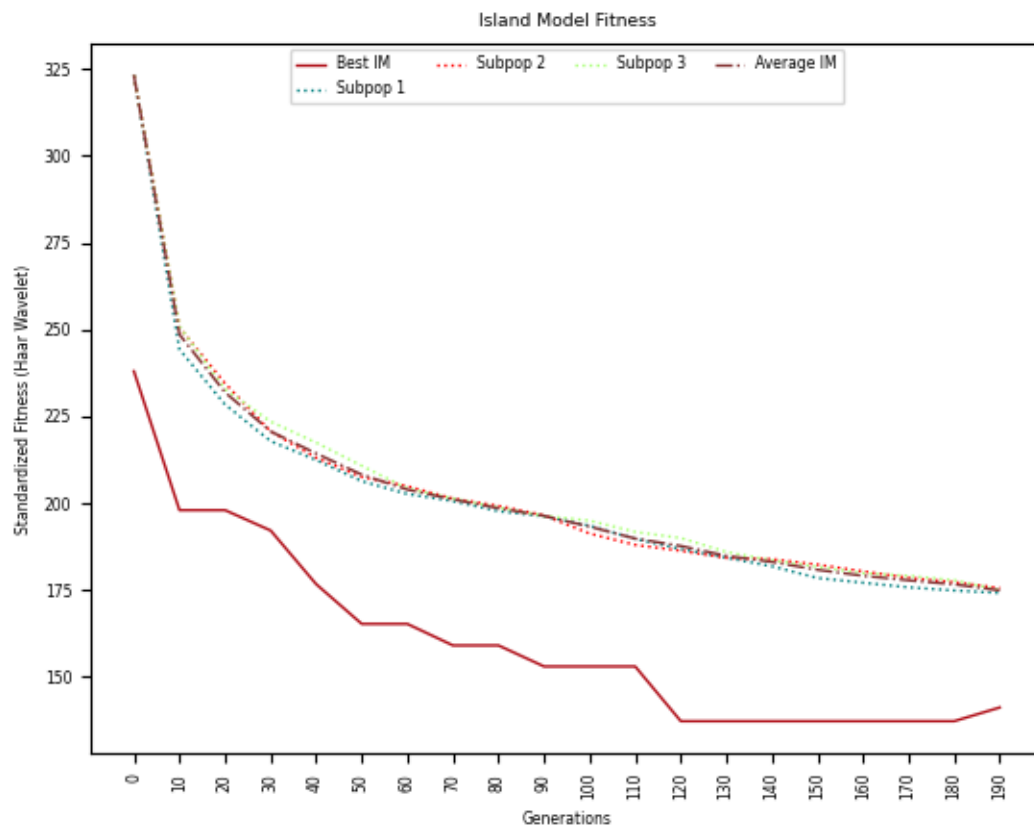


Figure 4.32: Best Fitness

The fitness plot in Figure 4.32 shows all subpopulations being very close to one another throughout all 200 generations of the runs. The first 10 generations seem to evolve the fastest before slowly evolving until generation 140 where it slows further before starting to plateau at generation 190.

4.2.4 Island-Model Results: Cartoon Smiling Face

Table 4.12: IM Image Results

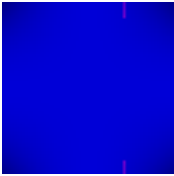
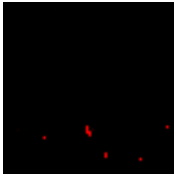
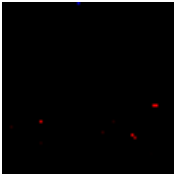
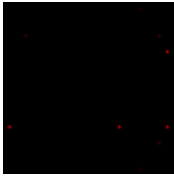
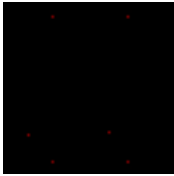
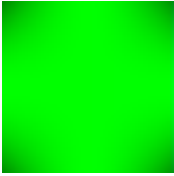
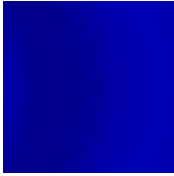
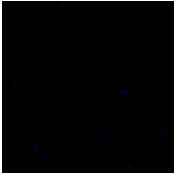
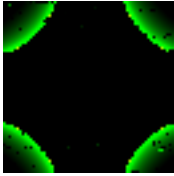
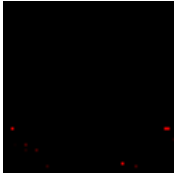


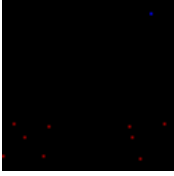


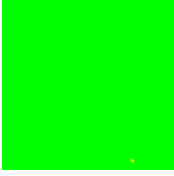
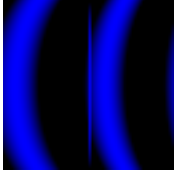
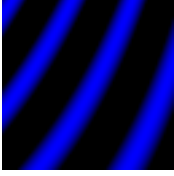

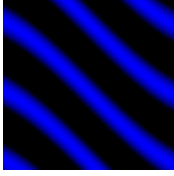


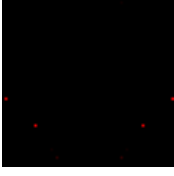
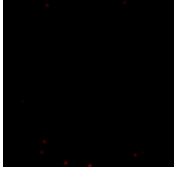
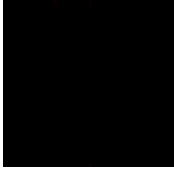
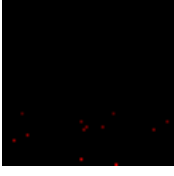
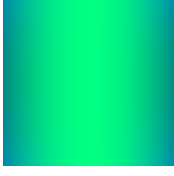
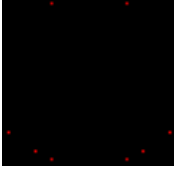
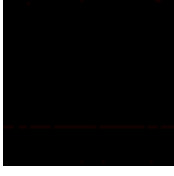

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				



Figure 4.33: Best Solution



Figure 4.34: Best Solution Coefficients

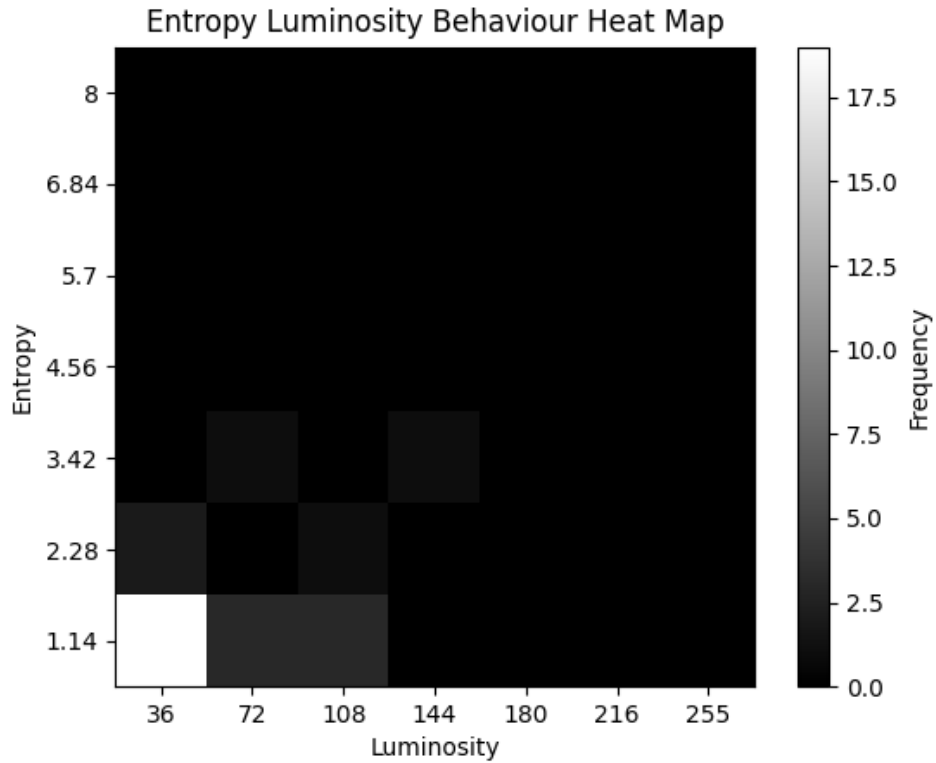


Figure 4.35: Entropy Luminosity Heat Map

Island-model evolution produces few but very interesting images from 30 runs in Table 4.12. There are three solutions of stripes with interesting curvatures. Runs 9 and 20 are also very interesting with patterns that focus on the corners of the image. The rest of the runs produced gradients or speckles across a black image.

The best solution from these runs is a gradient image in Figure 4.33. The coefficients used by this image have a general pattern similar to that of the coefficient placement to the target, however, it still falls very short when looking at the details. This solution fits into the entropy interval of 1.14-2.28 and luminosity of 36-72. Both entropy and luminosity of this best image falls short of the best solution with an entropy interval of 2.28-3.42 and luminosity 180-216.

The heat map concentrates on the same spot as the GP experiment with sparse runs producing solutions with the same entropy interval.

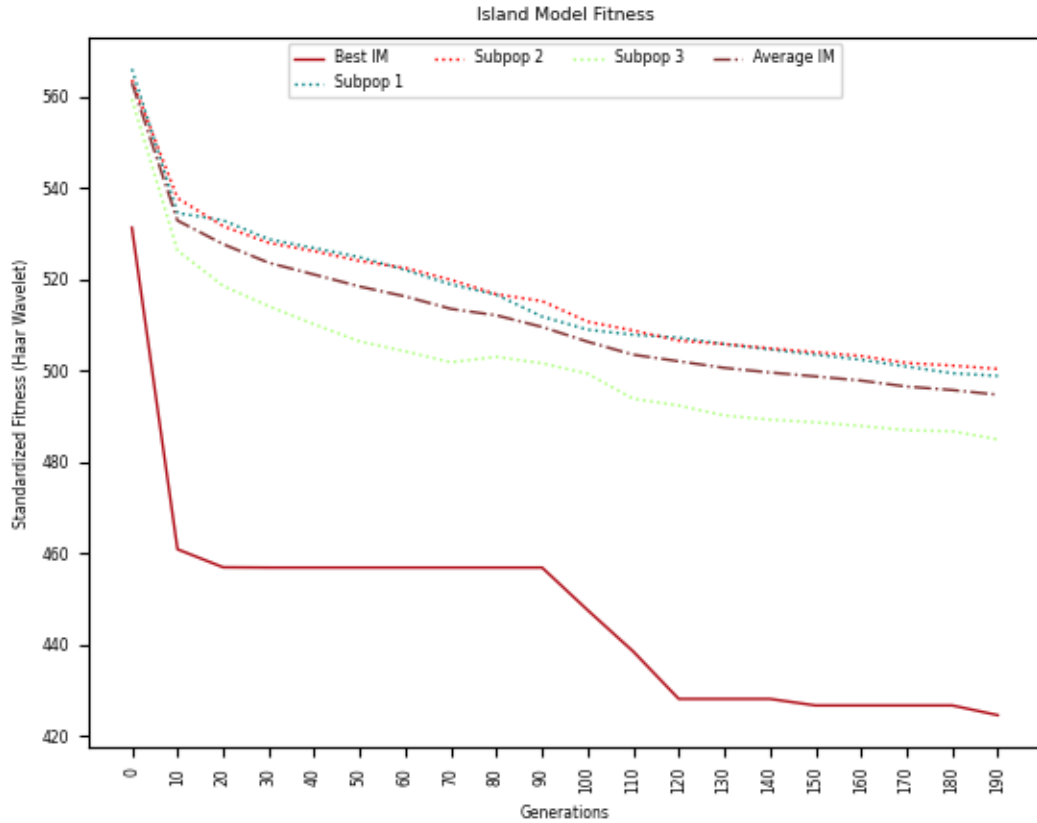


Figure 4.36: Best Fitness

The average fitness in Figure 4.36 is equivalent to that of the third subpopulation. Similar with all other island-model experiments, the spikes in fitness are due to frequent immigration and emigration. There is the most evolution happening within the first 10 generations before slowly evolving until generation 75 where the third subpopulation immigrates worse solutions and most likely emigrated its best solutions. It grabs its bearings at generation 105 before slowly evolving until generation 190. The best fitness quickly improves within the first 20 generations before plateauing until roughly generation 90. Afterwards it makes improvements with some stalls until generation 120 before plateauing with small increases until generation 195 before improving one last time.

4.2.5 MAP-Elites Results: Barcode

Table 4.13: Behaviour Intervals

Behaviour Name	Total Intervals	#1	#2	#3	#4	#5	#6	#7
Entropy	7	0	1.14	2.28	3.42	4.56	5.70	6.84
		1.13 $\bar{9}$	2.27 $\bar{9}$	3.41 $\bar{9}$	4.55 $\bar{9}$	5.69 $\bar{9}$	6.83 $\bar{9}$	8
Mean Luminosity	7	0	36	72	108	144	180	216
		35. $\bar{9}$	71. $\bar{9}$	107. $\bar{9}$	143. $\bar{9}$	179. $\bar{9}$	215. $\bar{9}$	255

This experiment does not involve the large dimension hypercube and opts in to a 2-dimensional layout. In Table 4.13 the entropy is calculated using the grey values of the entire image through the use of bit shifting the colour channels and luminosity is the average colour between the three channels. Since there are only two behaviours this will result in 49 total possible individuals within the MAP.

Table 4.14: ME Image Results using Target Figure 4.17

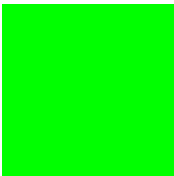

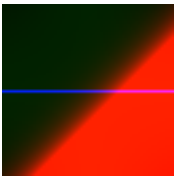

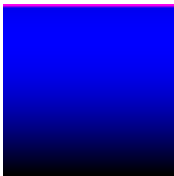

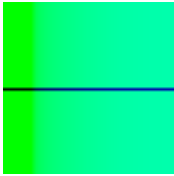
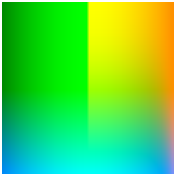
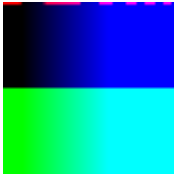

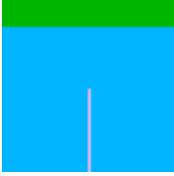
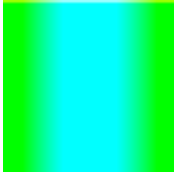
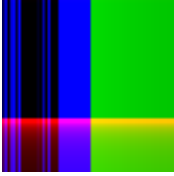
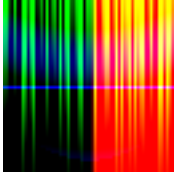
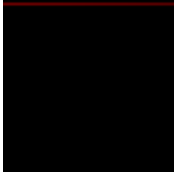
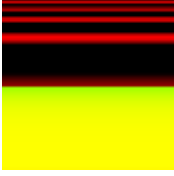
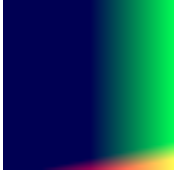
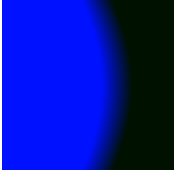
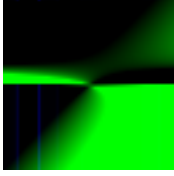
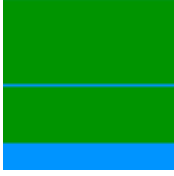

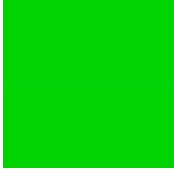
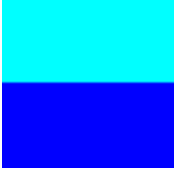

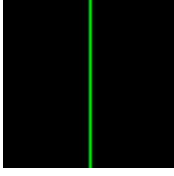

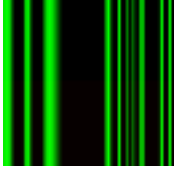
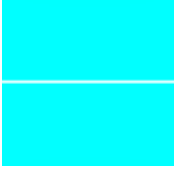
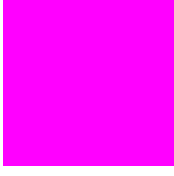
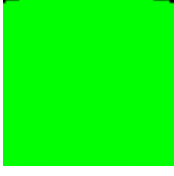
Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				



Figure 4.37: Best Solution



Figure 4.38: Best Solution Coefficients

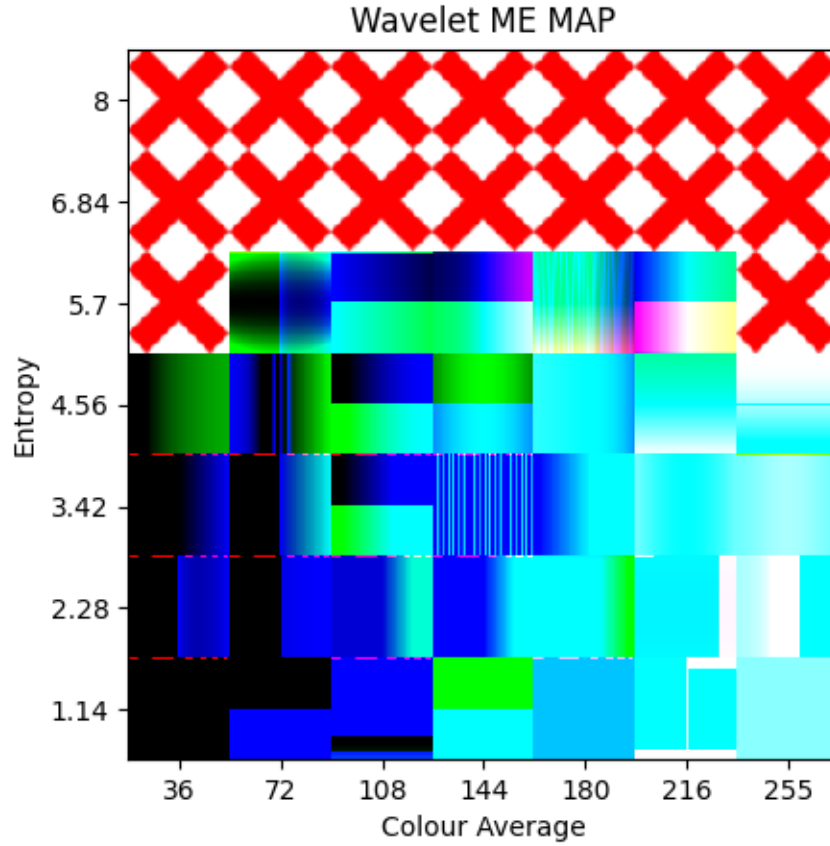


Figure 4.39: Best MAP using Target Figure 4.17

The best solutions in Table 4.14 typically have some stripe pattern within the image, however, very few resemble the barcode. Solutions 3 and 12 are the closest where solution 12 is very similar to a barcode and solution 3 is horizontal, but still resembles a barcode. The best solution found in Figure 4.37 is the best in fitness, however visibly, it would be argued as being worse than solutions 12 and 21. The coefficient image does show a similarity in the pattern of pixels for the highest valued coefficients in comparison to Figure 4.19.

Figure 4.39 resembles the MAP configuration which houses the best solution in Figure 4.37. It is observed that images with extremely high entropy in colour are not evolved which supports the evolution for a barcode target image. Visibly, there is a very close solution to the target image in the cell with a entropy of 3.42 and a colour

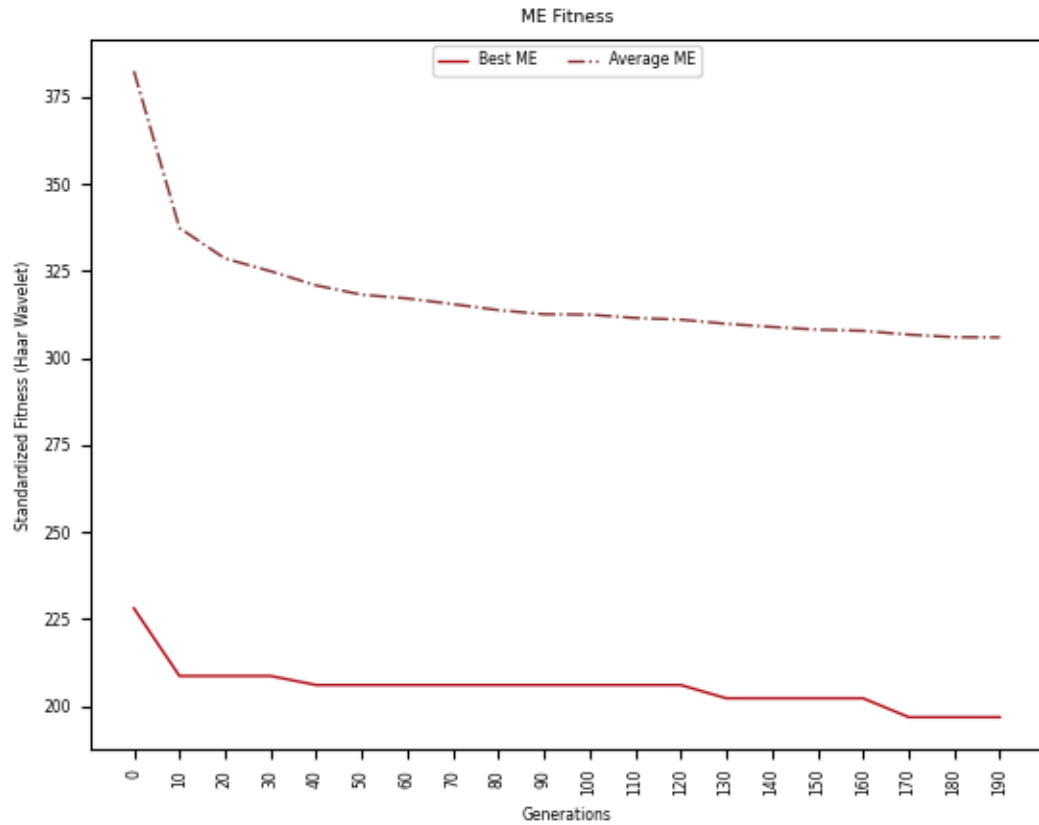


Figure 4.40: ME Fitness using Target Figure 4.17

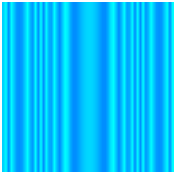
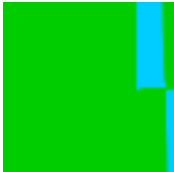
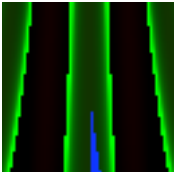
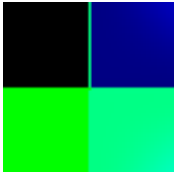
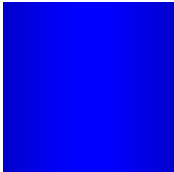
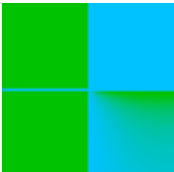
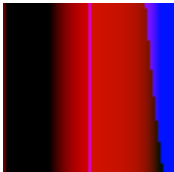
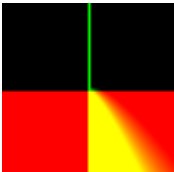
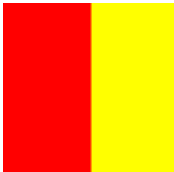
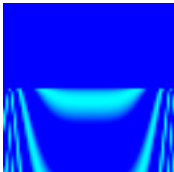
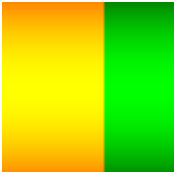
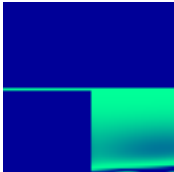

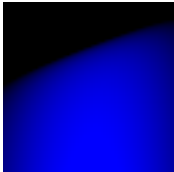
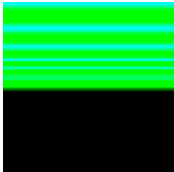
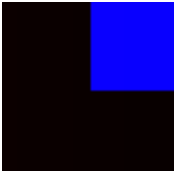

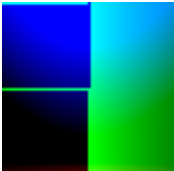

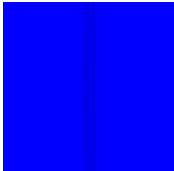

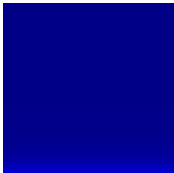
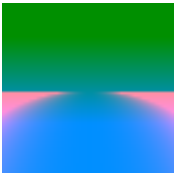
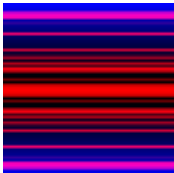



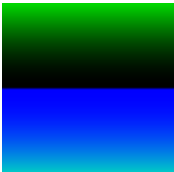
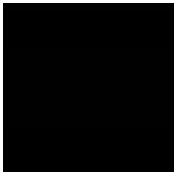
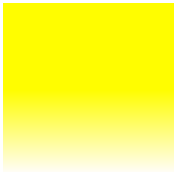
average, or luminosity, of 144.

The plot in Figure 4.40 averages the fitnesses of the entire MAP and the best individuals in the runs at each generation. The bulk of the evolution is roughly done from generations 0 to generation 25 before slowly evolving until a perceived convergence at generation 165. The best fitness improves the most within the first 15 generations before going on long plateaus with incremental improvements.

4.2.6 ME Results: Cartoon Smiling Face

This experiment utilizes the behaviour and interval set as Table 4.13.

Table 4.15: ME Image Results using Target Figure 4.17

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

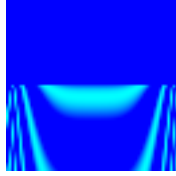


Figure 4.41: Best Solution



Figure 4.42: Best Solution Coefficients

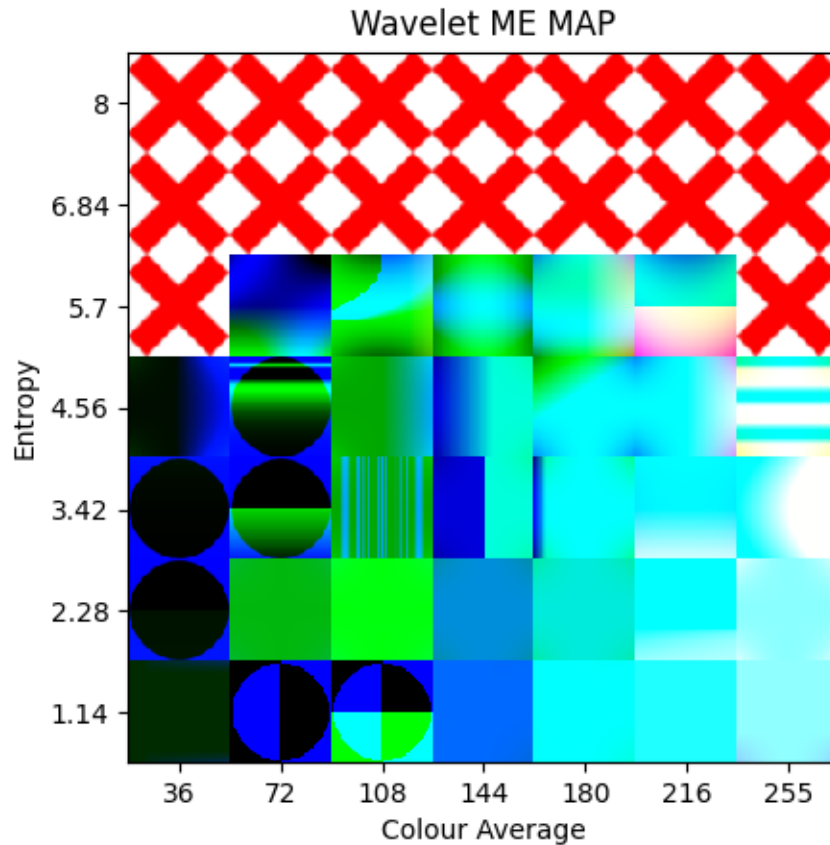


Figure 4.43: Best MAP using Target Figure 4.18

The best evolved solutions don't seem to have much in common with one another, some images seem to try and evolve the circle of the emoticon face while few others seem to try and evolve the smile. The rest of the evolved images are not seen as having evolved anything close otherwise. The best image in Figure 4.41 seems to be attempting to evolve the smile and the bottom of the face. The coefficients seem to be a little more expansive in comparison to the target which tends to be a little more

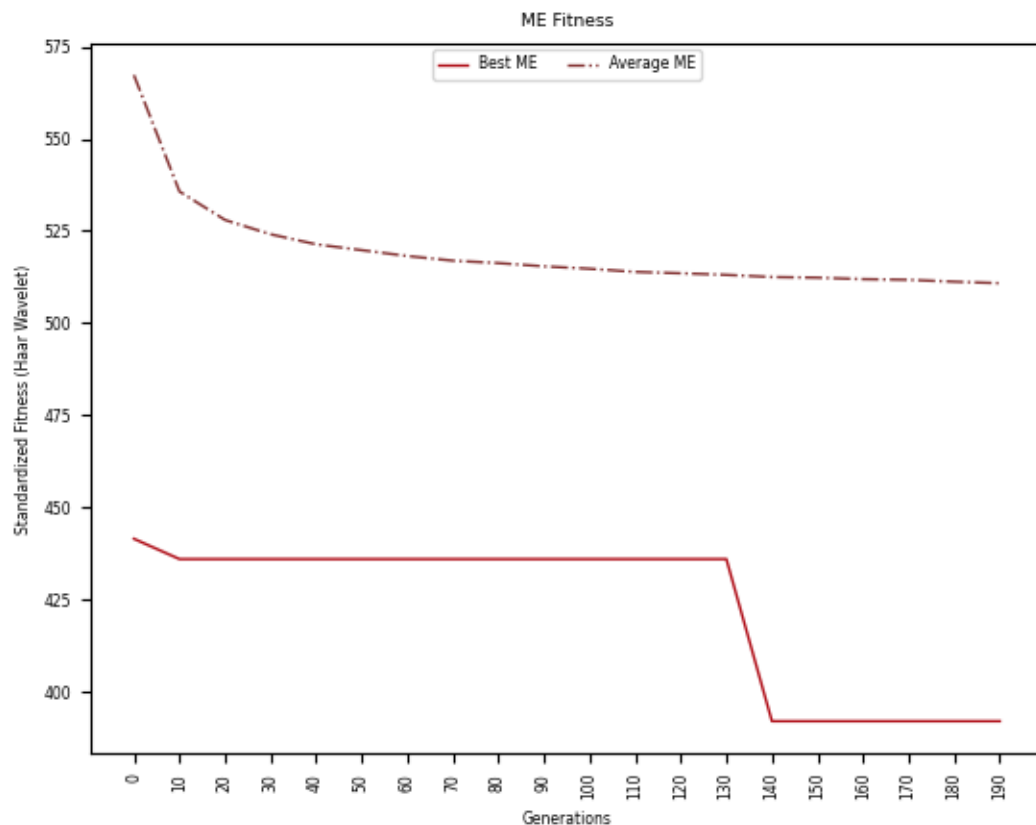


Figure 4.44: ME Fitness using Target Figure 4.18

compact in the positions of the pixels.

The fitness plot in Figure 4.44 depicts the average fitness of the MAP over the 30 runs as well as the best fitness. The fitness improves greatly until generation 5 for both the average and the best. The fitness improves slightly in the first 5 generations before finding nothing better until generation 135 where it then improves very sharply before plateauing until the end of the run.

4.2.7 DME Results: Barcode

Table 4.16: Wavelet DME Sub-MAPs

	Sub-MAP 1	Sub-MAP 2	Sub-MAP 3
Behaviour 1	Entropy	Entropy	Entropy
Behaviour 2	Luminosity	Luminosity	Luminosity

The DME configuration for this experiment utilizes the same behaviours for each island. This is intended to be equivalent to the ME algorithm in this experiment utilizing two behaviours as well. The total individuals remains the same, 147, since the number of intervals are the same. Utilizing the same behaviours, in this experiment DME produces 3 times as many individuals as ME since there are 3 islands.

Table 4.17: DME Image Results using Target Figure 4.17

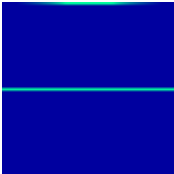
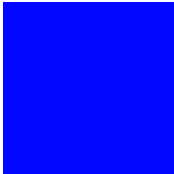
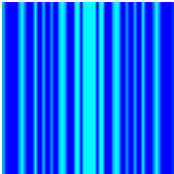
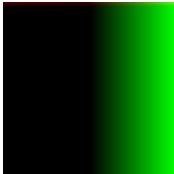
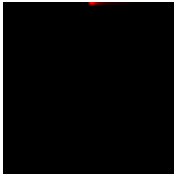
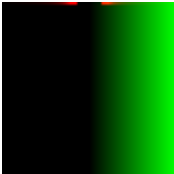

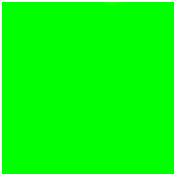

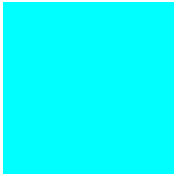



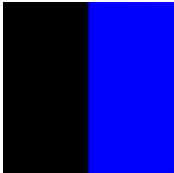
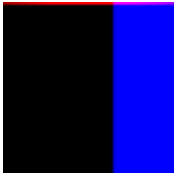
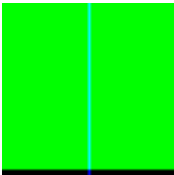
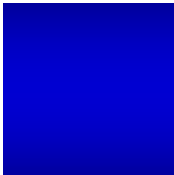
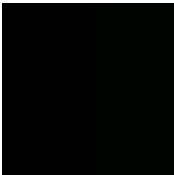
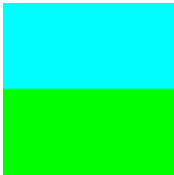
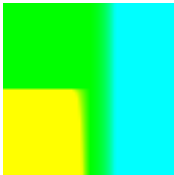
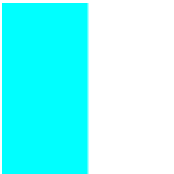
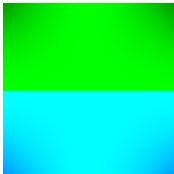
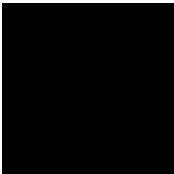

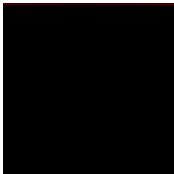
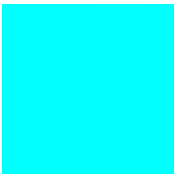
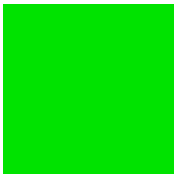
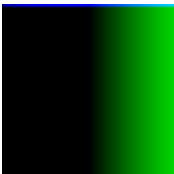
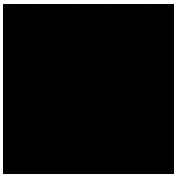

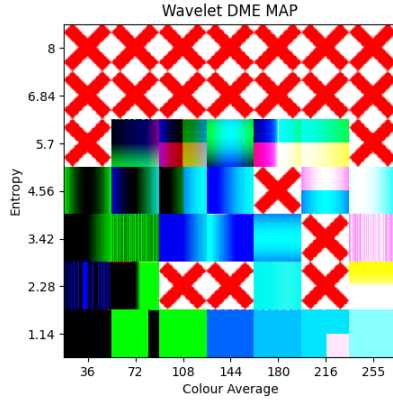
Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				



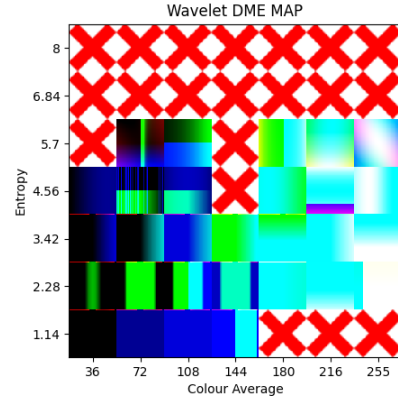
Figure 4.45: Best Solution



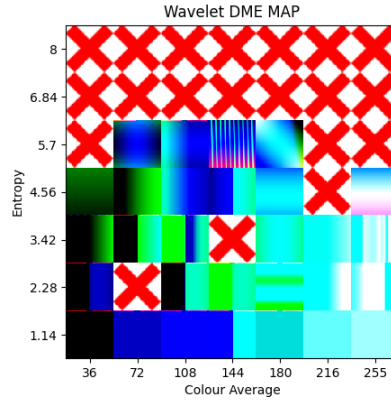
Figure 4.46: Best Solution Coefficient



(a) Entropy and Luminosity Island MAP 1



(b) Entropy and Luminosity Island MAP 2



(c) Entropy and Luminosity Island MAP 3

Figure 4.47: Wavelet DME Visualization

The best solutions from the runs have all evolved solutions which are visibly seen as very poor in Table 4.17. Bar solution 13 which does resemble a barcode the other solutions are mostly flat coloured images with slightly different pixel colours at the top, gradients, or two shaded images. The best solution in Figure 4.45 is an image that is a flat colour with different colours at the top of the image.

Figure 4.47 is the configuration from run 30 which houses the best solution in Figure 4.45. The first and third submaps do have solutions evolved which resemble barcodes or things which are close to barcodes. In the first configuration solutions in the cells of entropy 3.42 and colour average 72 and entropy 3.42 and colour average

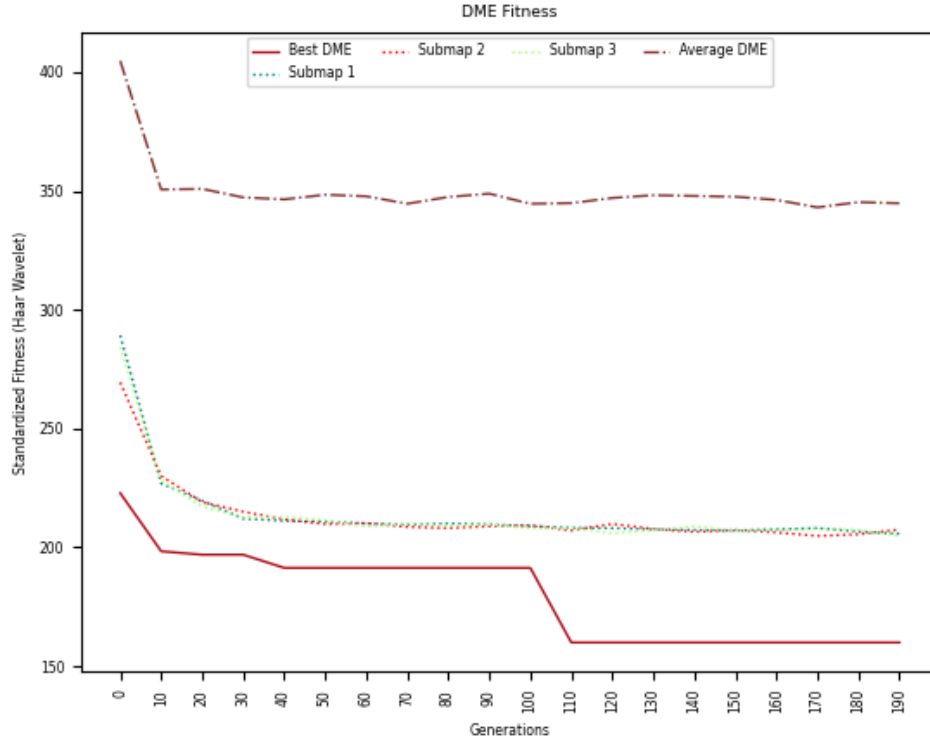


Figure 4.48: DME Fitness

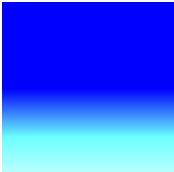
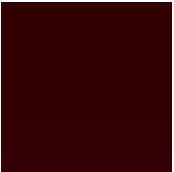
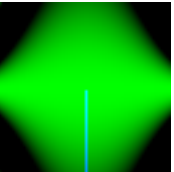
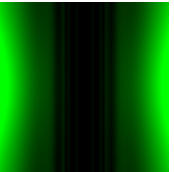
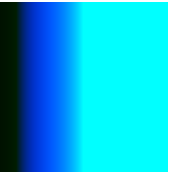

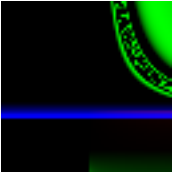
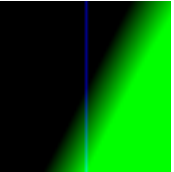
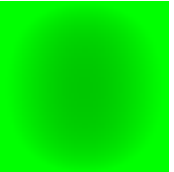
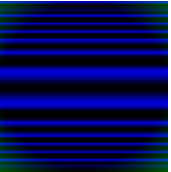
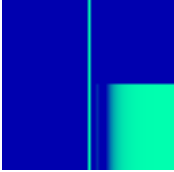
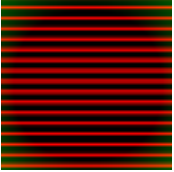
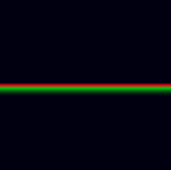
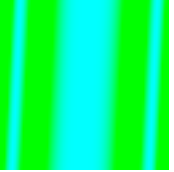
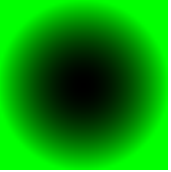
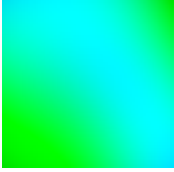
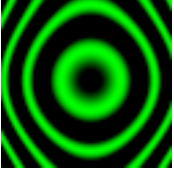
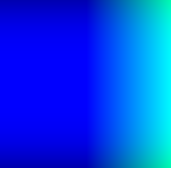
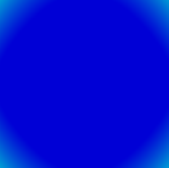
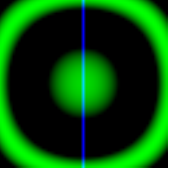
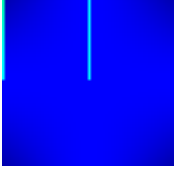
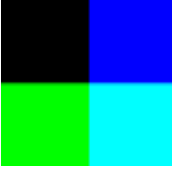
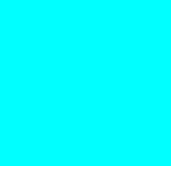

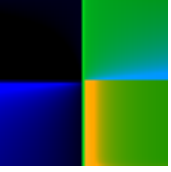

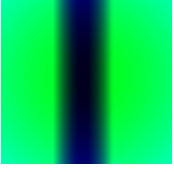
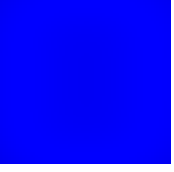


255 closely resemble barcodes. In the third configuration the cell with entropy 5.7 and colour average 144 and entropy 3.42 and colour average 255 resemble it as well. Although the former solution in the third configuration is very significantly different and loosely resembling the target in comparison to the latter.

Figure 4.48 includes the average fitness, the best of the sub-maps as well as the best fitness on each generation averaged over 30 runs. The average fitness continues to be very jagged with a slight downward trajectory. The submaps all have a similar growth curve and fitness values exchanging positions with one another. The best fitness improves rapidly within the first 10 generations and in generations 110-115 while the rest are fairly flat.

4.2.8 DME Results: Cartoon Smiling Face

This experiment utilizes the same configuration and behaviour set as Table 4.16.

Table 4.18: DME Image Results using Target Figure 4.18

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

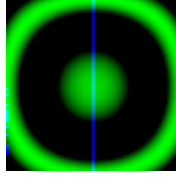
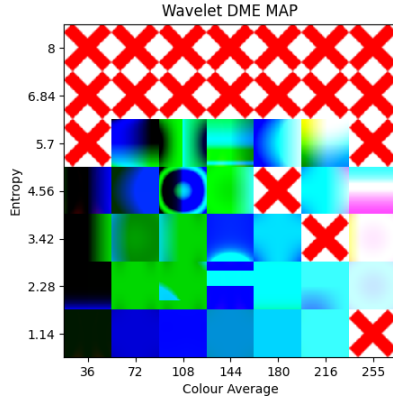


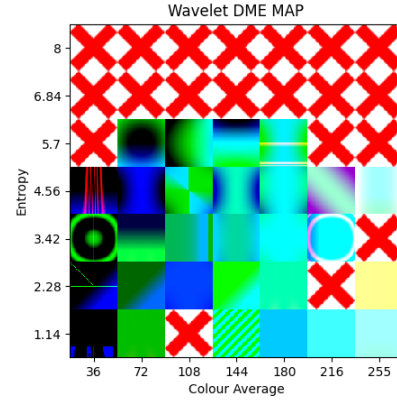
Figure 4.49: Best Solution



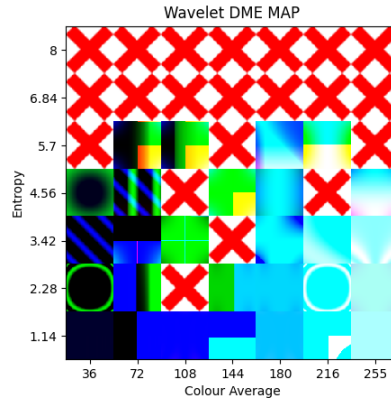
Figure 4.50: Best Solution Coefficient



(a) Entropy and Luminosity Island MAP 1



(b) Entropy and Luminosity Island MAP 2



(c) Entropy and Luminosity Island MAP 3

Figure 4.51: Wavelet DME Visualization

Some of the best solutions evolved do evolve some aspects of the difficult target image of the cartoon smiling face. In multiple solutions the circle being as the outline of the face. However, there are still many runs which produce nonsensical solutions which don't seem to have aspects similar to the target image. There are some very

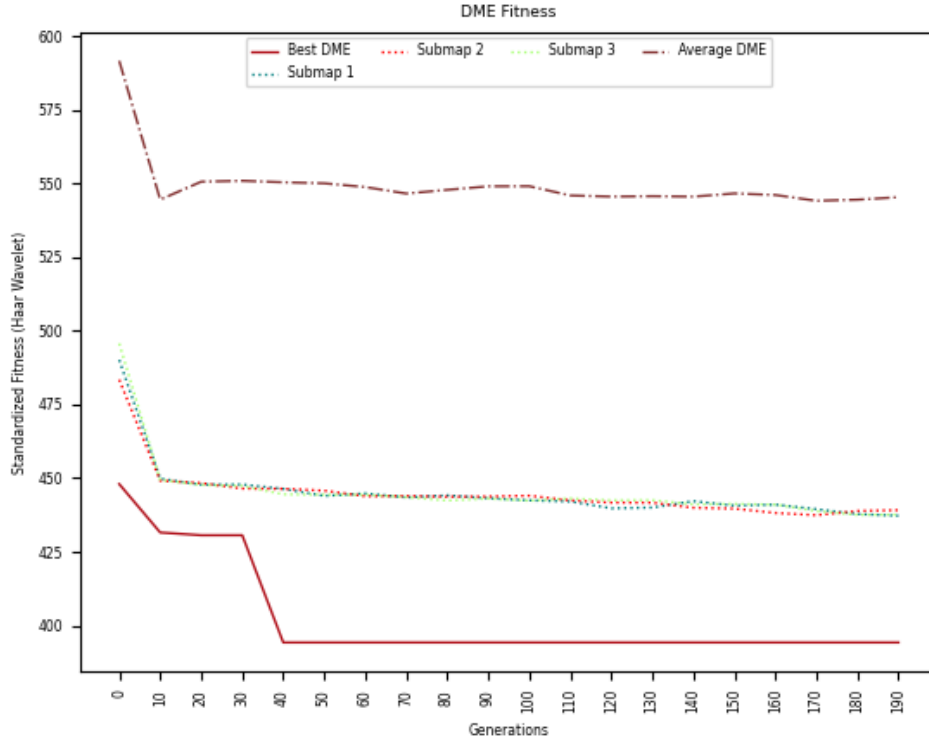


Figure 4.52: DME Fitness

interesting solutions that have been evolved like in run 8 and 19. The best solution evolves a circle within an outlined circle which is similar to that of the target image. Each sub-map configuration has solutions which resemble the circles of the target image. Most of the solutions evolved within the MAP are still quite poor visually in comparison to the target.

Figure 4.52 shows very fast evolution for the submaps in the first 5 generations before slowly improving its fitness until a perceived convergence at generation 180. The average fitness maintains the same as the with the barcode target image, a higher fitness while being very jagged. The best fitness improves slightly within the first 15 generations before improving greatly in generations 30-35. Afterwards it stays flat finding no better values.

4.2.9 Discussion

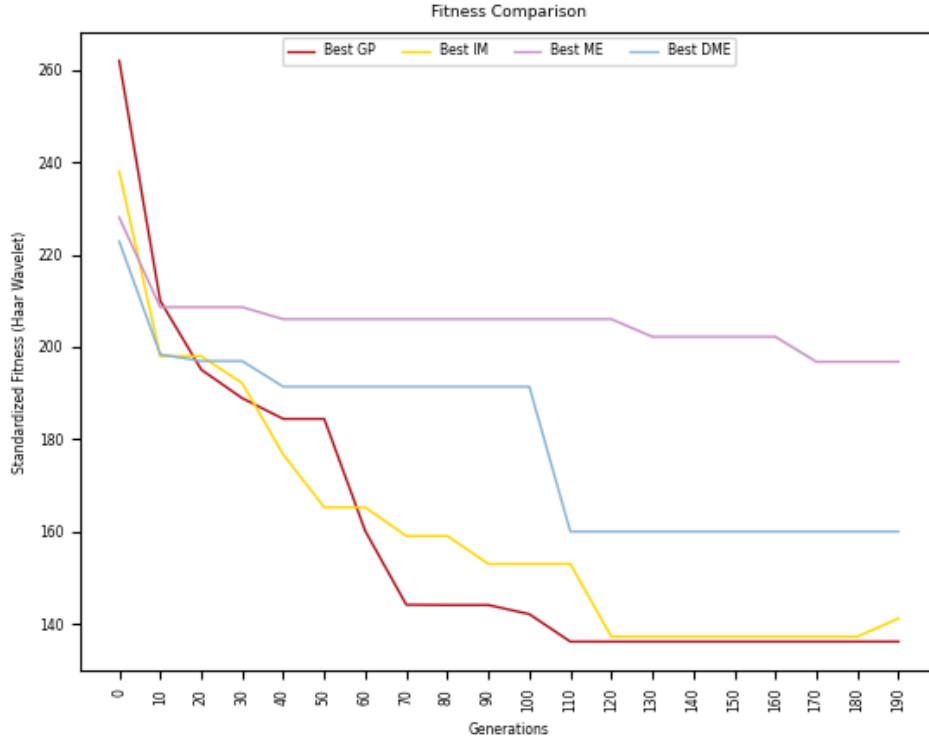


Figure 4.53: Barcode Fitness Comparison

GP and island-model evolution outperform both ME and DME. Around generation 150 GP and island-model appear to come to roughly the same fitness value. DME outperforms ME by a much wider margin than the previous two competitors despite ME being slightly ahead during initialization. The margin between the vanilla variants in comparison to the MAP variants is not widely significant but still quite wide which is opposite to the visual deductions made in each of the algorithms results. ME seems to improve in short bursts while DME improves in very large chunks.

Figure 4.54 is the behaviour box plot for the luminosity, or colour average. Figure 4.55 is the behaviour box plot for entropy. These behaviours were calculated from the best solutions from the 30 runs for all the algorithms. Vanilla GP and island-model optimized the same types of solutions for the majority of runs since outliers are not

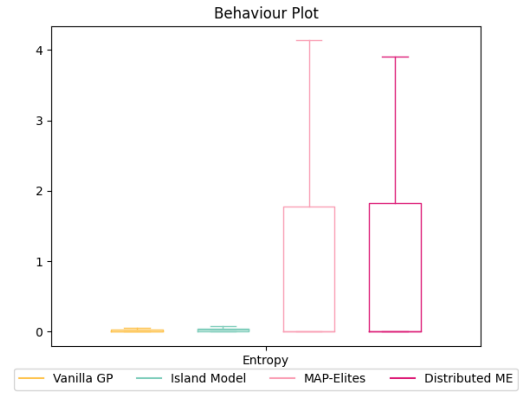
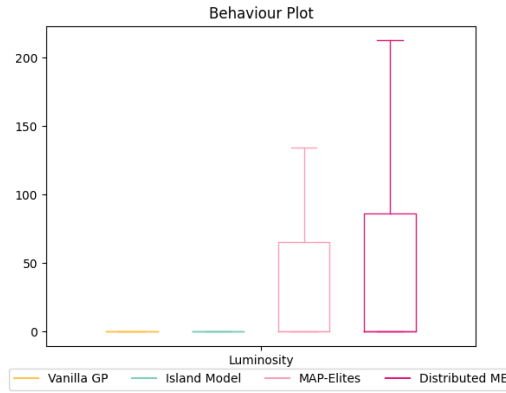


Figure 4.54: Barcode Luminosity Box Plot Figure 4.55: Barcode Entropy Box Plot

shown in these box plots. Whereas ME and DME had very different entropies and luminosities.

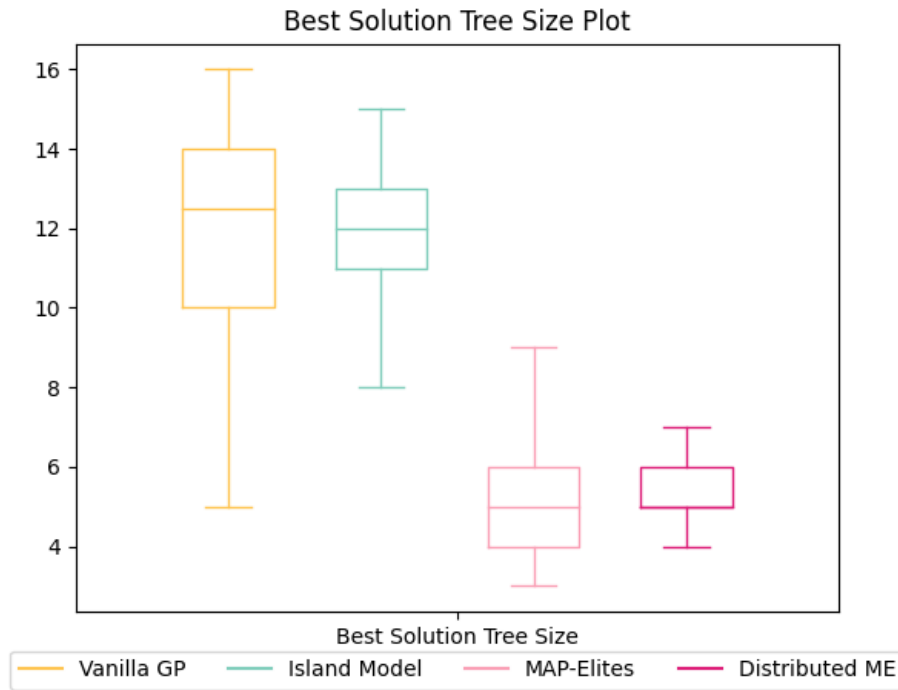


Figure 4.56: Barcode Tree Depth Box Plot

Figure 4.56 shows a box plot with the depths of the best final solutions obtained from each of the algorithms using the 30 runs. The median of vanilla GP and island-model final solution depths hover around 12-14 where ME and DME hover around 4-6.

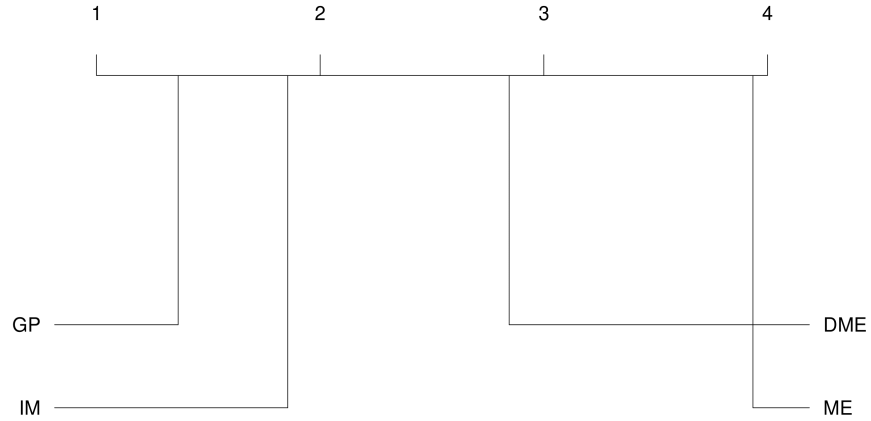


Figure 4.57: Barcode Critical Difference

Vanilla GP has quite a wide range from the first to third quartile going from roughly 5-16, island-model approximately 8-15, ME 1-9, and DME 4-7. On average, ME and DME are producing much smaller trees in comparison to GP and island-model.

All following statistics have been formulated within R utilizing the Friedman test and Shaffer correction using a threshold of $p < 0.05$, with the null hypothesis:

$$H_0 : \theta_{GP} = \theta_{IM} = \theta_{ME} = \theta_{DME}$$

The algorithms ran on the experiment using the barcode as a target image in Figure 4.57 are statistically significant due to the lack of a bar bisecting multiple algorithms. Therefore, the null hypothesis is rejected and the medians of the algorithms are not equal to one another. The same deductions made through the fitness plot are equivalent here due to the usage of the fitness values where GP ranks first, followed by island-model then DME and ME. One interesting observation is that GP and IM are ranked quite close to each other whereas DME is closer to IM than it is to ME.

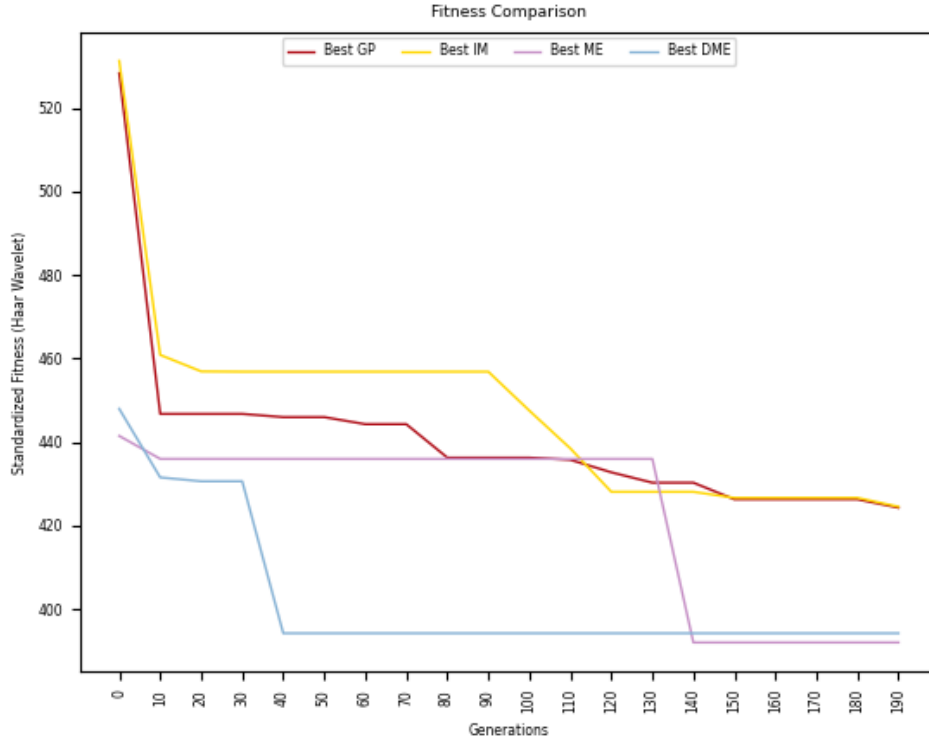


Figure 4.58: Cartoon Smiling Face Fitness Comparison

The fitness comparison in Figure 4.58 compares the best fitness from each algorithm run using the cartoon smiling face target image. With this specific basic language and implementation of utilizing haar wavelet transformations the ME variants outperform the vanilla variants with DME performing the best then ME, island-model, and finally GP. From the start the ME variants have better fitness values in comparison to the vanilla variants. IM and GP remain fairly close throughout all the generations while DME and ME have an interesting relationship. ME quickly finds a good fitness which persists until the end of the run while DME takes more than half of the generations before slightly outperforming ME. IM outperforms GP for almost two thirds of the runtime before having very close fitness to it swapping back and forth.

Figures 4.59 and 4.60 are the behaviour box plots for entropy and luminosity using

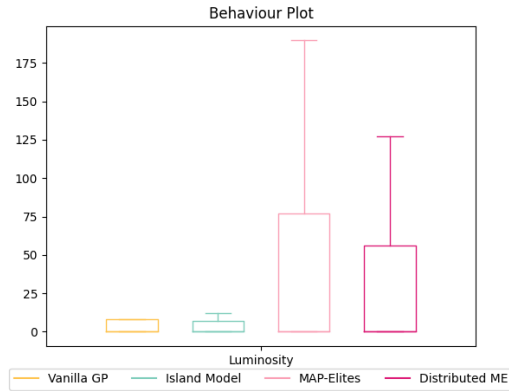


Figure 4.59: Luminosity Box Plot

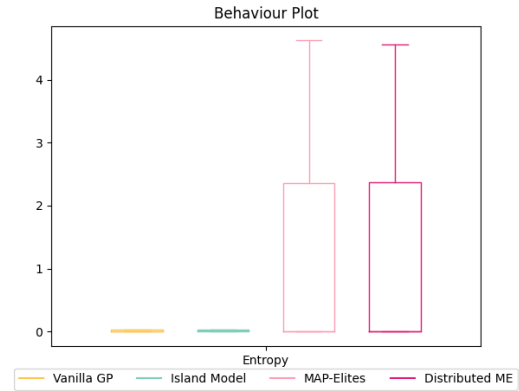


Figure 4.60: Entropy Box Plot

the cartoon smiling face target image. The same pattern occurs where vanilla GP and island-model evolve to the same small range of behaviours where ME and DME are much more diverse just using the best solutions. Using this target image, ME produced more solutions with a more varied luminosity than DME where the barcode was the opposite.

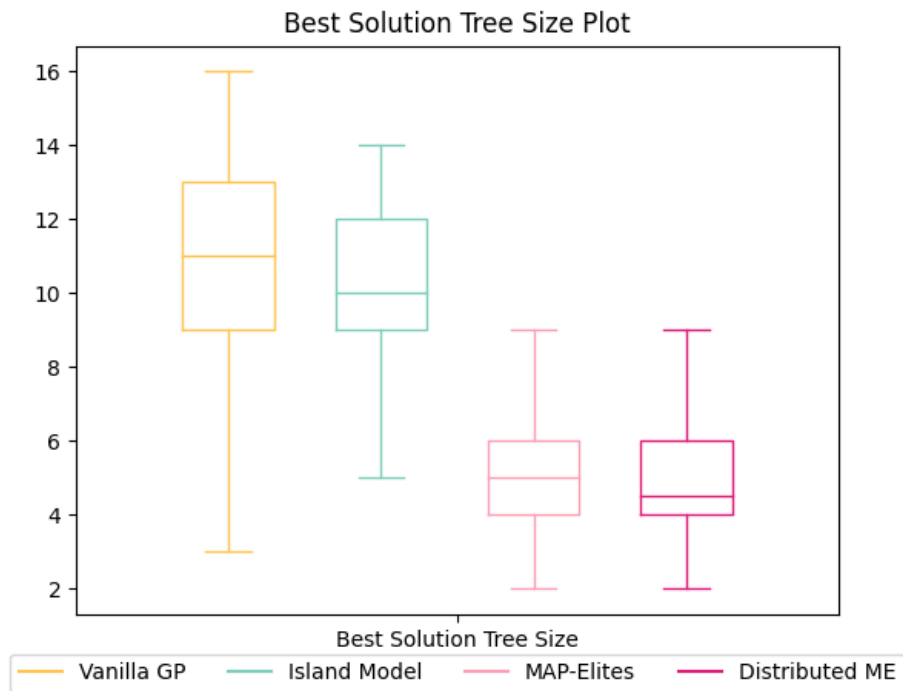


Figure 4.61: Cartoon Smiling Face Tree Depth Box Plot

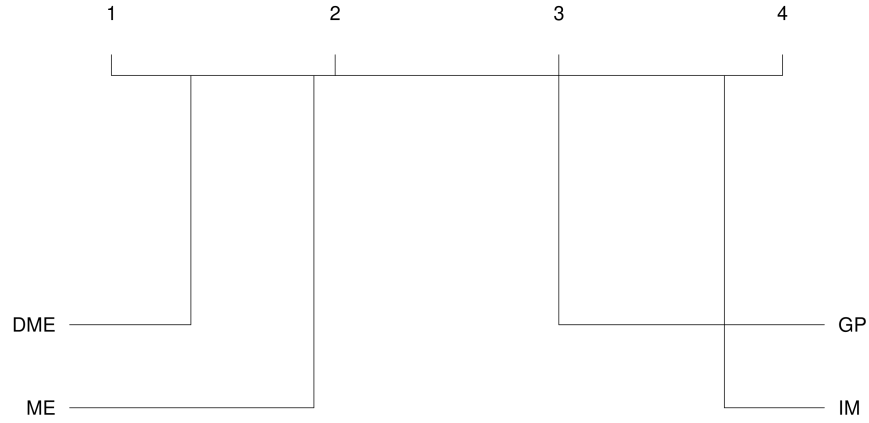


Figure 4.62: Cartoon Smiling Face Critical Difference

Figure 4.61 is the box plot of the algorithm's best solution's tree depth using the cartoon smiling face target image. Vanilla GP and island-model have medians hovering around 10-11 where ME and DME are roughly between 4 and 5. Vanilla GP continues to have the larger range having the first quartile minimum be around 3 and it's third quartile max at 16. Island-model has a higher minimum first quartile around 5 and a lower maximum third quartile around 14. ME and DME are very similar here both having maximum third quartiles around 9 and minimum first quartiles around 2.

All following statistics have been formulated within R utilizing the Friedman test and Shaffer correction using a threshold of $p < 0.05$, with the null hypothesis:

$$H_0 : \theta_{GP} = \theta_{IM} = \theta_{ME} = \theta_{DME}$$

The algorithms ran on the experiment utilizing the cartoon smiling face target image are statistically significant due to the lack of a bar bisecting multiple algorithms. Therefore, the null hypothesis is rejected and the medians of the algorithms are not equal to one another. The rankings on the critical difference diagram are equivalent to that of the fitness comparison graph where DME comes out on top, ME in second, GP in third, and IM trailing behind in the fourth rank. Each of these ranks are fairly

cut and dry given the locations of the algorithms relative to the rank positions. DME and ME are slightly closer as observed in comparison to GP and island-model.

When using this specific fitness function and GP configuration, a more complicated target image seems to produce more valuable results than a simple target image on vanilla GP and island-model algorithms. ME and DME both provided very interesting and much more potentially similar solutions for both sets of target images. The MAP visualizations add more on top of the best solutions from the runs being visually more valuable.

4.3 Wavelet SSIM Experiment

This experiment is a comparison between two similar fitness functions, wavelet transformations and SSIM. These two functions aim to identify main aspects of the image rather than specific details. Wavelet focusing on coefficients for each pixel, and SSIM calculating aspects of each window across an entire image to come up with the score. The behaviours in the MAP-Elite variant algorithms will be the same as the wavelet experiment. Figure 4.63 will be the target image for this experiment. The ME algorithm will be utilizing the same behaviours, behaviour interval total, and interval values as Table 4.8. The DME algorithm will also be following the previous wavelet experiment with a configuration of Table 4.16 This rough drawing of a sunset will serve as a somewhat difficult target to evolve for in order to compare the two fitness functions. Figure 4.64 is what the fitness function derived as the most important pixel locations for the coefficients of this sunset target image.

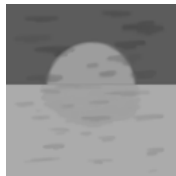


Figure 4.63: Target Image



Figure 4.64: Sunset Coefficients

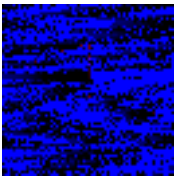

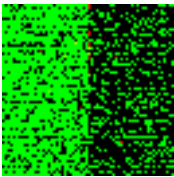
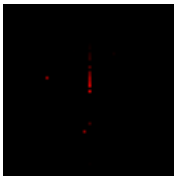
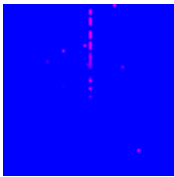
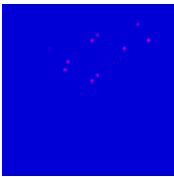
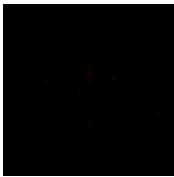
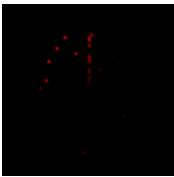
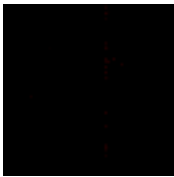
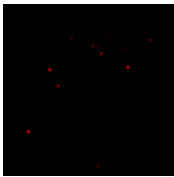
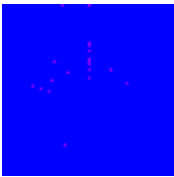
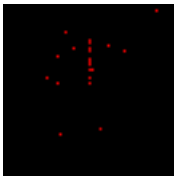
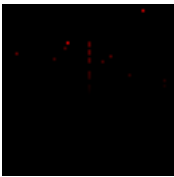
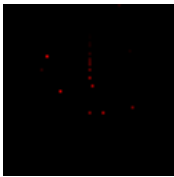
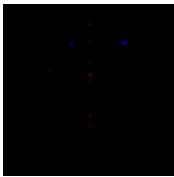
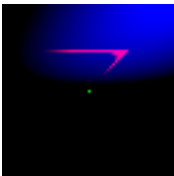
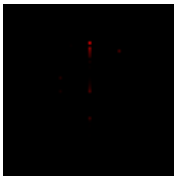
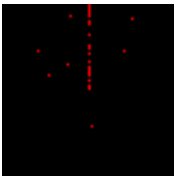
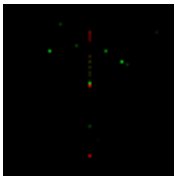
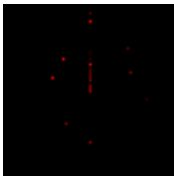
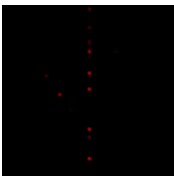
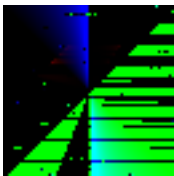


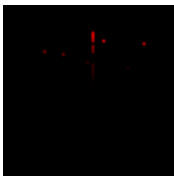
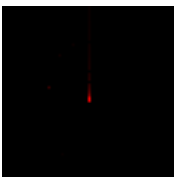
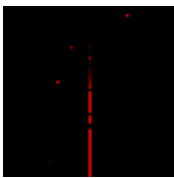
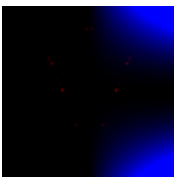
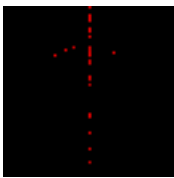
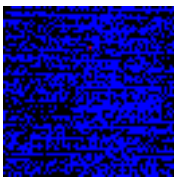
Fitness Function 1	Haar Wavelet Coefficient Comparison	Ranking x highest value coefficients and comparing the respective indice's rank.
Fitness Function 2	SSIM	Calculating the SSIM over every window in the image.

Table 4.19: Fitness Function and Behaviours

Table 4.19 highlights both fitness functions that are being utilized in this experiment. As previously mentioned, SSIM will be utilizing 8x8 pixel windows over the entire image of size 128x128 pixels in a sliding fashion. Once every window's SSIM value has been calculated an average of this index will be made which will result in the fitness value.

4.3.1 Vanilla GP Results: Wavelet

Table 4.20: GP Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

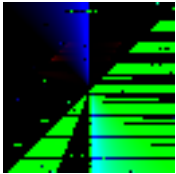


Figure 4.65: Best Solution



Figure 4.66: Best Solution Coefficient

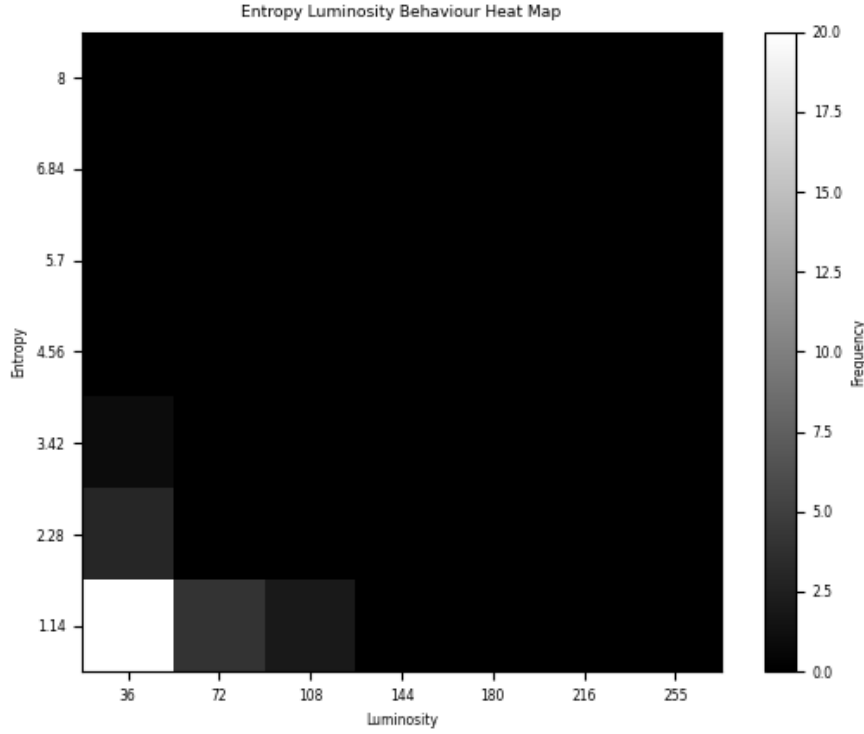


Figure 4.67: Entropy Luminosity Heat Map

Table 4.20 represents the best solutions from the 30 runs using the wavelet fitness on the sunset target image. The majority of solutions evolved a solid coloured image with some speckles of pixels or lines lightly scattered throughout. There are some solutions where there are interesting patterns instead of a flat background or interesting geometry or patterns made instead of seemingly random pixels placed throughout.

The best solution in Figure 4.65 shows a very interesting pattern on the bottom right half of the image. A line bisecting the middle of the image is also seen which can be interpreted as a horizon line except in the y dimension. When comparing behaviours, the target image occupies the cell with entropy 2.28-3.42 and luminosity 108-144. This solution occupies entropy 1.14-2.28 and luminosity 0-36. In terms of entropy it is quite close, however falls flat in entropy showing it values the luminosity much less with wavelet fitness. The coefficients it chose as the most important are

also quite similar in pattern with the sectioning however the target's coefficients are much more sparse than this solution.

The diversity of the best solutions generated in Figure 4.67 show a majority of evolution in the 0-1.14 entropy and 0-36 luminosity without much exploration from there. Marginal amounts of solutions are found 1 entropy interval above with a luminosity of 0-36 and 2 luminosity intervals further at the 0-1.14 entropy.

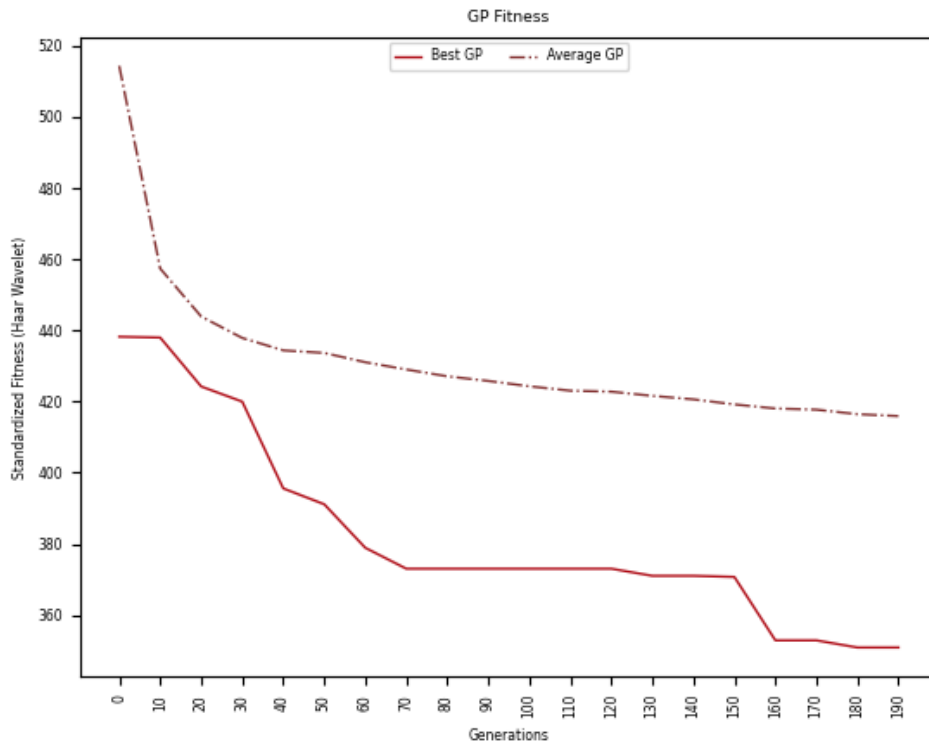
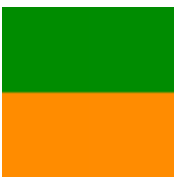
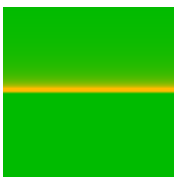
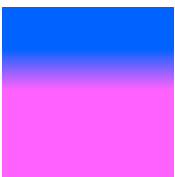

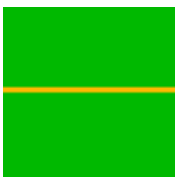
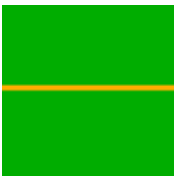
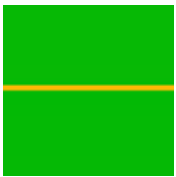
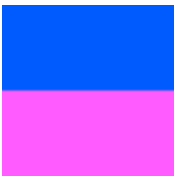
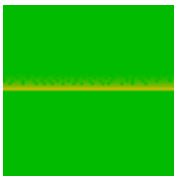
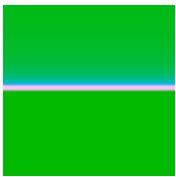
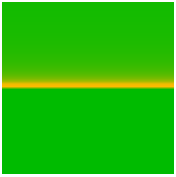
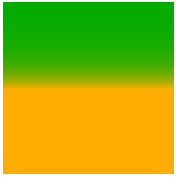


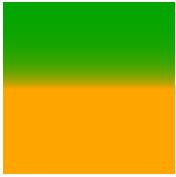
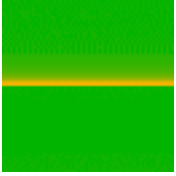

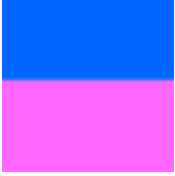
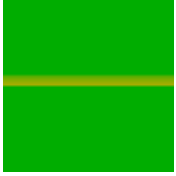
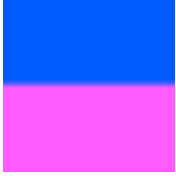
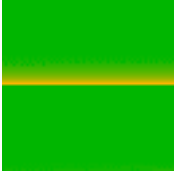
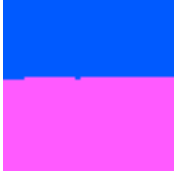
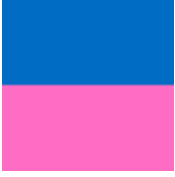
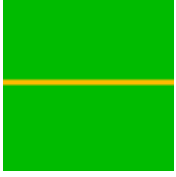
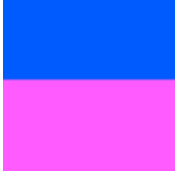
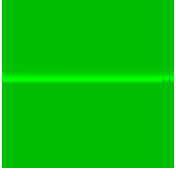
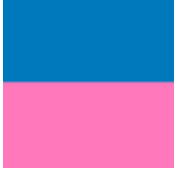


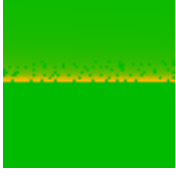


Figure 4.68: GP Fitness

The average fitness in Figure 4.68 shows a very smooth evolution throughout the generations. Most of the evolution is done in the first 30 generations before slowing down until generation 170. Past generation 170 it is seen that the fitness only makes extremely marginal improvements, almost if not already at convergence. The best fitness has a tumultuous improvement from generations 10 to 70 before being mostly flat until generation 150 before it makes another significant improvement.

4.3.2 Vanilla GP Results: SSIM

Table 4.21: GP Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

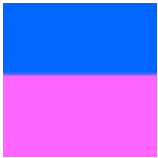


Figure 4.69: Best Solution

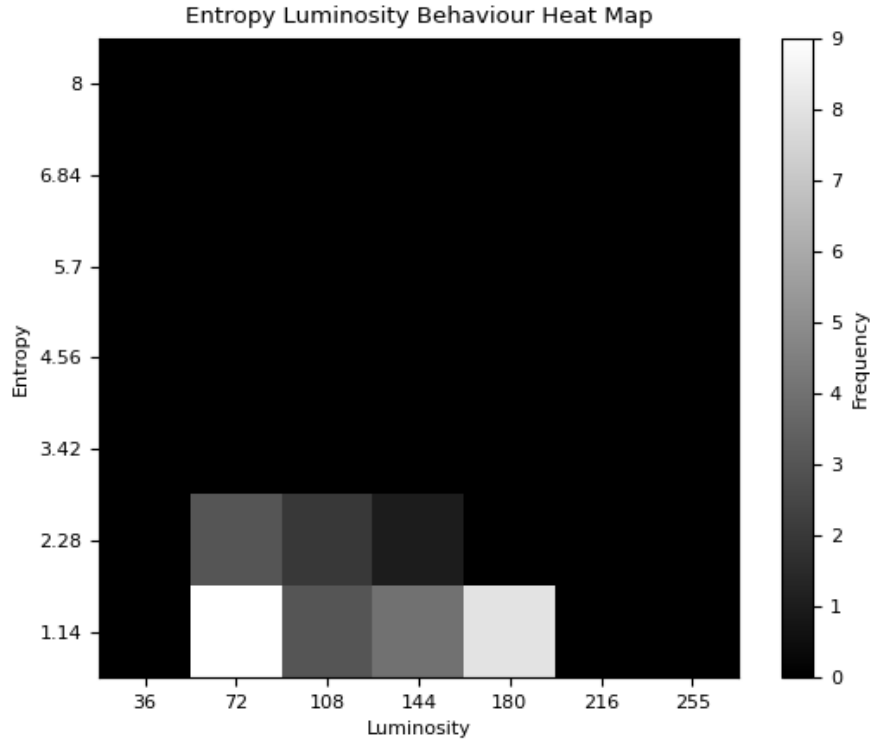


Figure 4.70: Entropy Luminosity Heat Map

The best solutions from 30 runs utilizing SSIM as the fitness function in Table 4.21 all evolve to one general type of solution. This is the horizon line separating the sun and sky of the target image. Some of these results got closer having orange and red colours, and others had gradients, but overall, none close to the target image.

The best solution in Figure 4.69 has a solid split of colours as the sky and sea with a slight blur on the horizon line. When comparing behaviours, the target image has an entropy interval of 2.28-3.42 and a luminosity of 108-144. This best solution has an entropy of 0-1.14 and a luminosity of 144-180. It is relatively close in terms of luminosity but fairly far from entropy.

Figure 4.70 is the diversity heat map for GP using SSIM. There is a decent amount of diversity in the cells shown. GP has found low entropy and mid-values of luminosity to be the best over the 30 runs as seen from the cluster of cells.

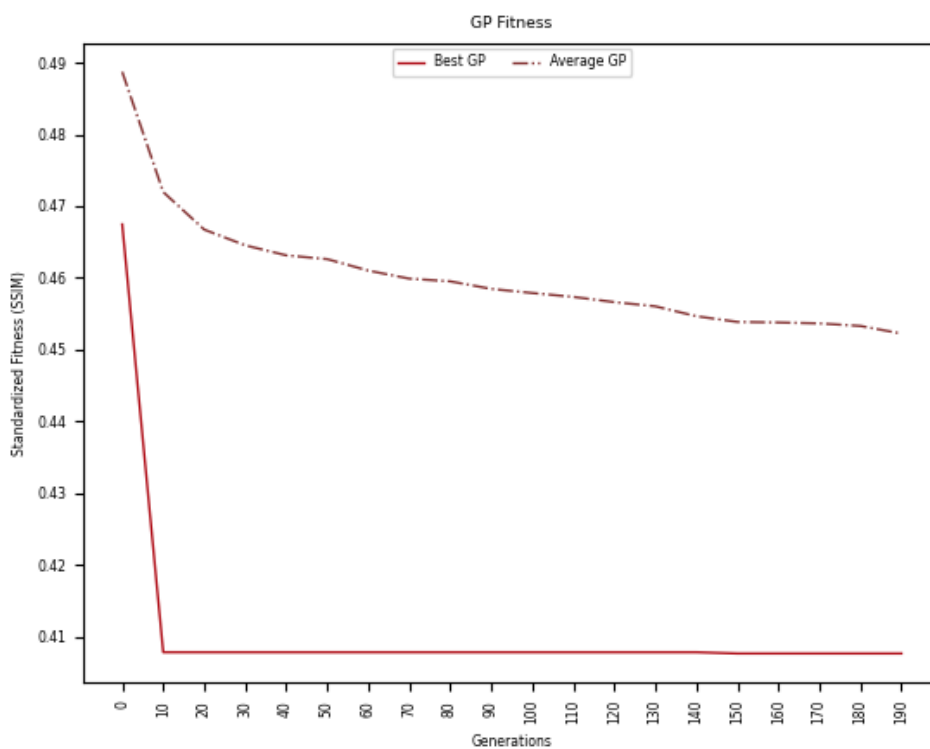
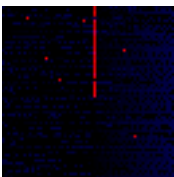

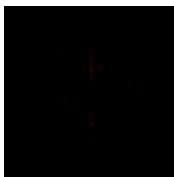
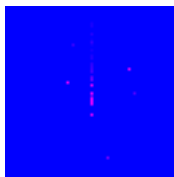
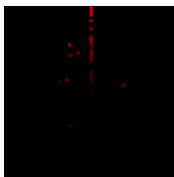
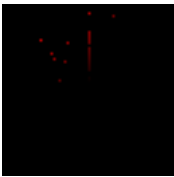
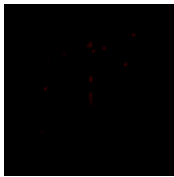
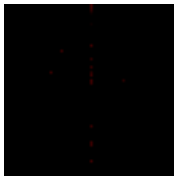
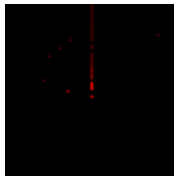
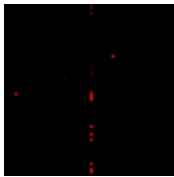
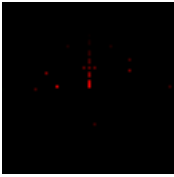
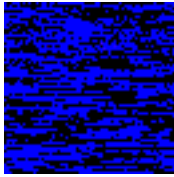
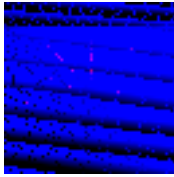
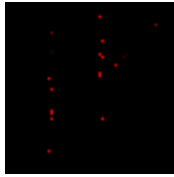
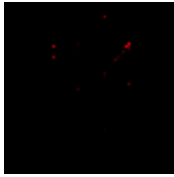


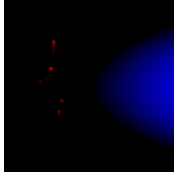
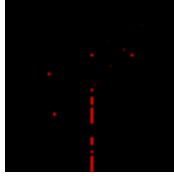
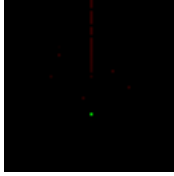
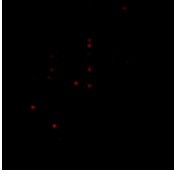
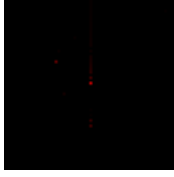
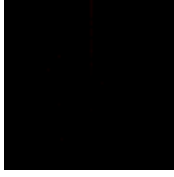
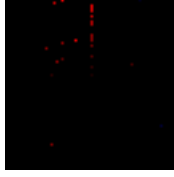
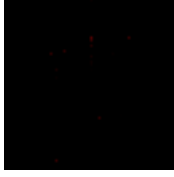

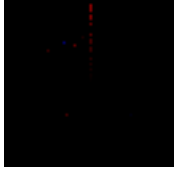
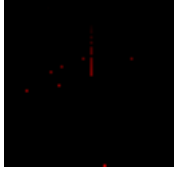

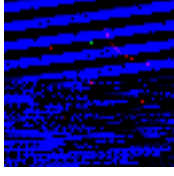


Figure 4.71: GP Fitness

The fitness plot in Figure 4.71 shows the majority of evolution in the average being done in the first 20 generations of the runs. After these 20 generations it slowly evolves with some plateaus until generation 185 where a convergence seems to happen. The best fitness is found within the first 15 generations.

4.3.3 Island-Model Results: Wavelet

Table 4.22: IM Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

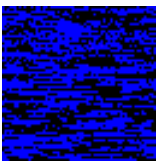


Figure 4.72: Best Solution



Figure 4.73: Best Solution Coefficient

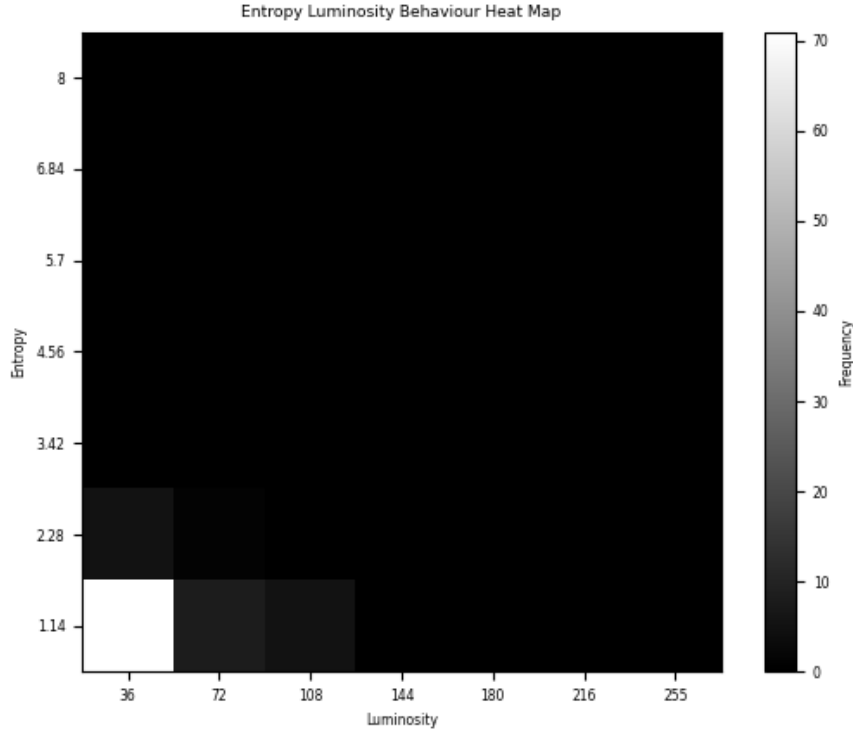


Figure 4.74: Entropy Luminosity Heat Map

The majority of best solutions presented in Table 4.22 consist of a pure black image with some arbitrary red pixels, or some red pixels attempting to create a line, or a combination of the two. There are 5 solutions which are notably different than the aforementioned pattern, runs 9, 15, 16, 19, and 30. Runs 9, 15, and 30 seem to try and evolve patterns which can be interpreted as an attempt at the clouds and their reflections. Run 16 has some blue protruding from the side of the image and 19 opts for a blue background.

The best run in Figure 4.72 is one of the more interesting solutions which has the diagonal stripes from run 15 as well as the interesting patterns from run 9. On the behaviour interval side, it has an entropy of 0-1.14 and a luminosity of 36-72 which is a far cry from the target with an entropy of 2.28-3.42 and luminosity of 108-144. The coefficients used for this solution are much more organized and clumped in a

systematic manner in comparison to the target.

The diversity heat map in Figure 4.74 shows a lack of diversity in the exact same format as with vanilla GP in Figure 4.67.

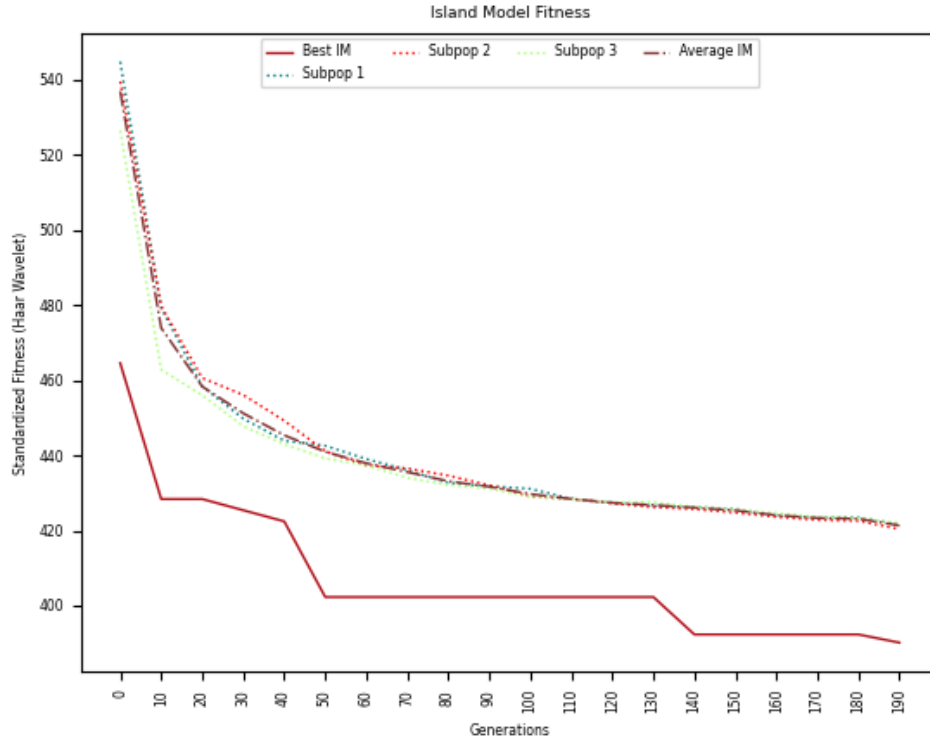
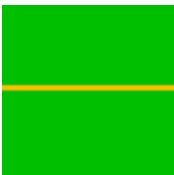

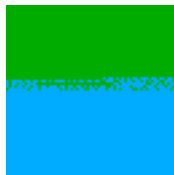
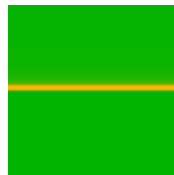

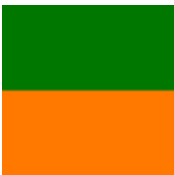



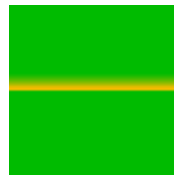

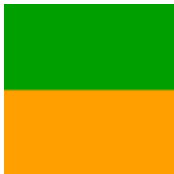

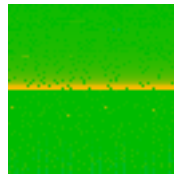
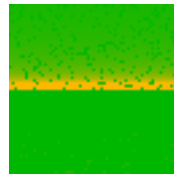
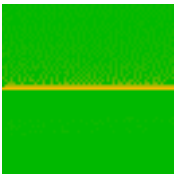
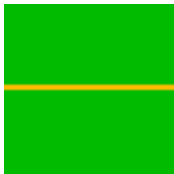



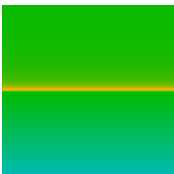


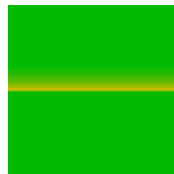




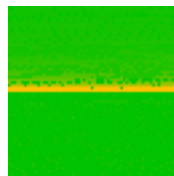



Figure 4.75: IM Fitness

The average fitness and subpopulation's best fitness in Figure 4.75 are very close to each other improving at a similar rate. It is seen that subpop 3 performs better in the first 40 generations before being very similar in fitness to the rest. The best fitness improves in chunks from generations 0-10, 25-45, 130-135 and 190-200. The greatest improvement being at the beginning 10 generations.

4.3.4 Island-Model Results: SSIM

Table 4.23: IM Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

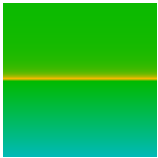


Figure 4.76: Best Solution

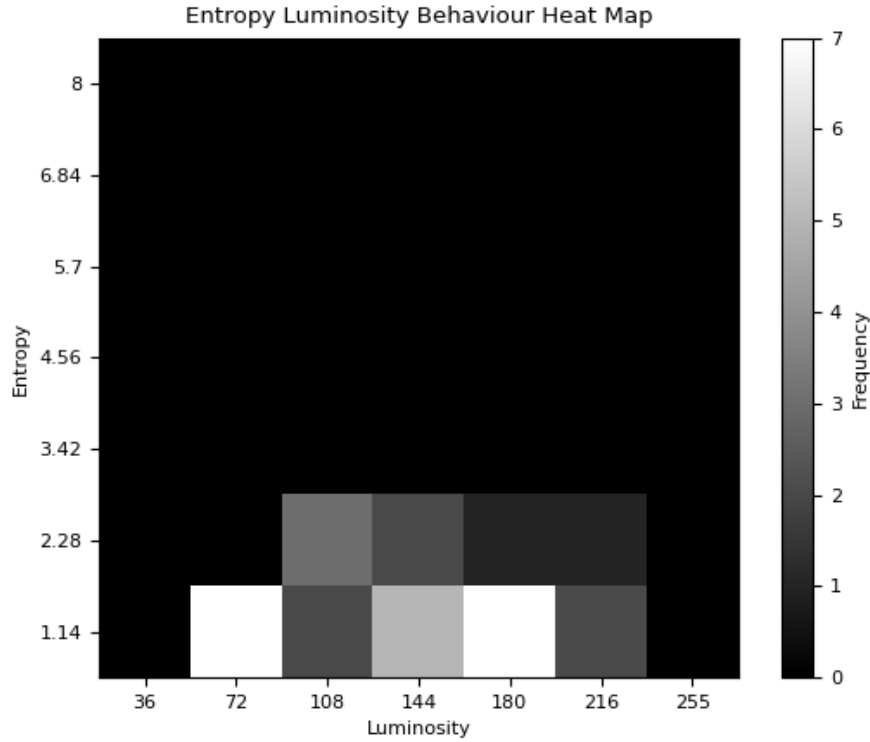


Figure 4.77: Entropy Luminosity Heat Map

The best solutions in Table 4.23 mostly evolve to a blurred horizon line with a slight gradient or some flat images. There are some interesting variations on this with sparse pixels around the horizon area making the gradient less clean.

The best solution from the 30 runs in Figure 4.76 opts in favour of the blurred horizon line with a slight gradient on it upwards as well as on the bottom.

The behaviour heat map in Figure 4.77 reflects the same general findings as the behaviour heat map in Figure 4.70 by having low entropy, however, the luminosity ranges much more which may be in part due to the increase of solutions generated by island-model.

The third subpopulation identified in the fitness plot in Figure 4.78 outperforms the other subpopulations. The scale is not by much, but visually it is quite apparent. Subpopulation 1 and 2 seem to improve in a similar way with the second subpopula-

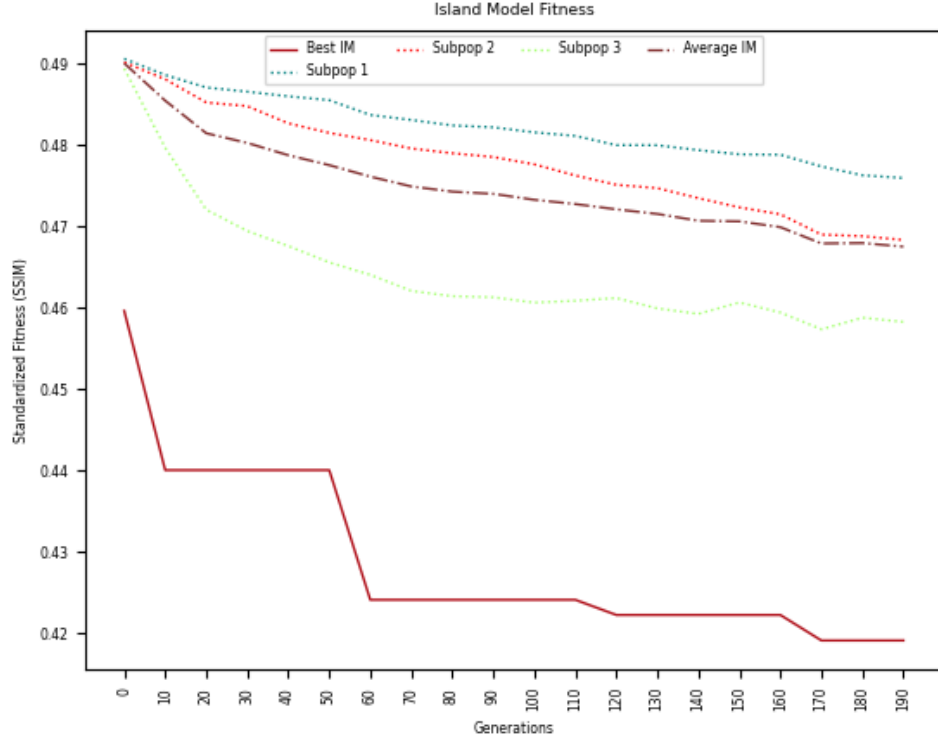


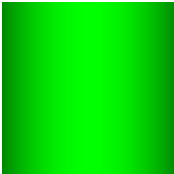
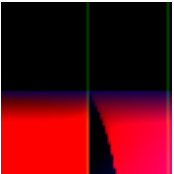
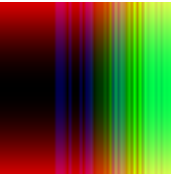
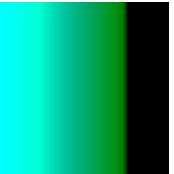
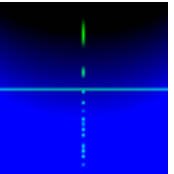

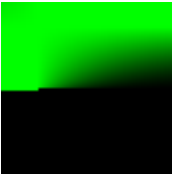

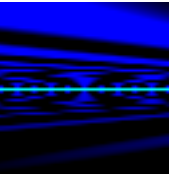
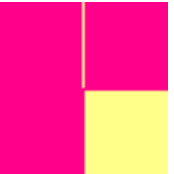
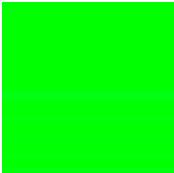

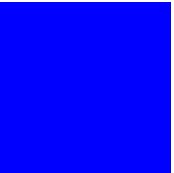

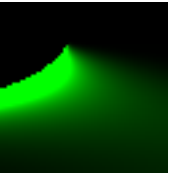

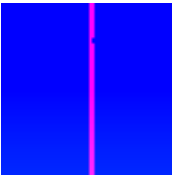
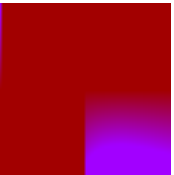
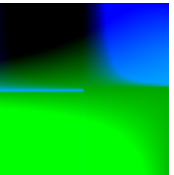
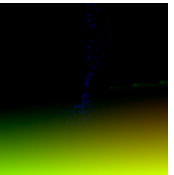
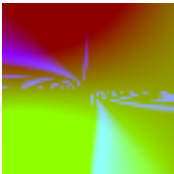
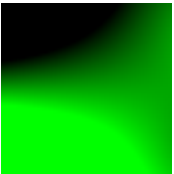
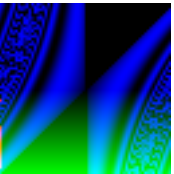



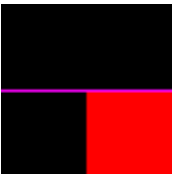
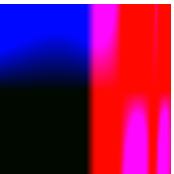
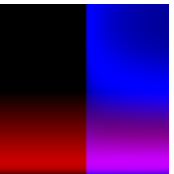
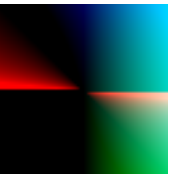
Figure 4.78: IM Fitness

tion doing better. Subpopulation 3 quickly evolves quicker until generation 60 before slowing down onward for the rest of the generations. All subpopulations are seen with some jaggedness due to the immigration and emigration, the most intense being the third subpopulation due to the difference in fitness. The average fitness lies on the lower end due to the intense margin between the third subpopulation and the rest. The best fitness improves very sharply in two spots, the first 5 generations and around the 50th generation. Otherwise, it makes marginal improvements while being mostly flat.

4.3.5 MAP-Elite Results: Wavelet

The MAP configurations remain the same as the previous two experiments in Table 4.13.

Table 4.24: ME Image Results using Target Figure 4.63

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

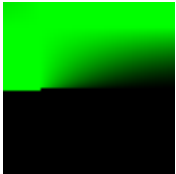


Figure 4.79: Best Solution



Figure 4.80: Best Solution Coefficient

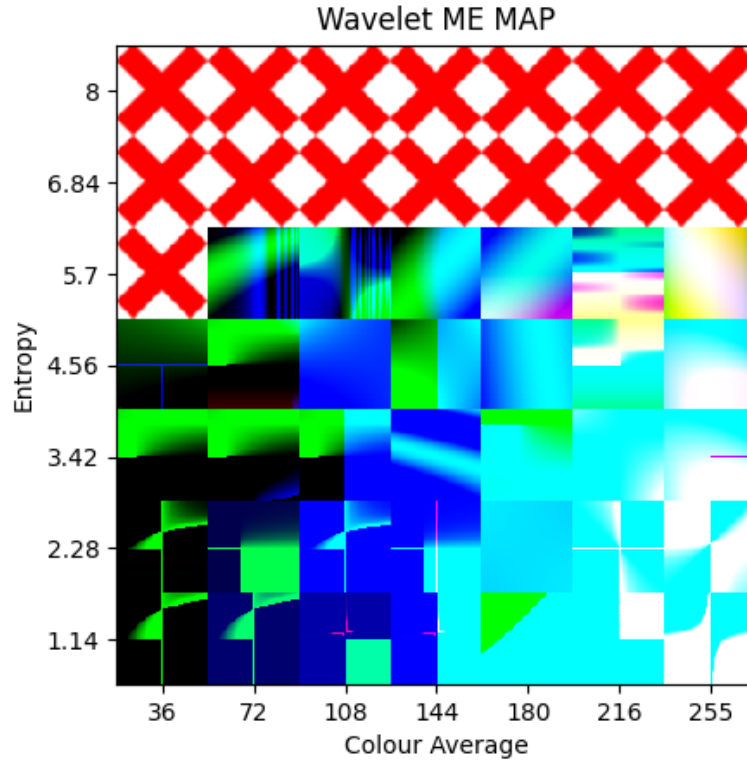


Figure 4.81: MAP Visualization

The compilation of best solutions from the 30 runs in Table 4.24 shows a wide variety of solutions. There are a few solutions that have identified that there is a line down the middle representing horizon line. Some of these runs correctly identified it as being horizontal while others opted for a vertical line instead. On top of this there is interesting geometry and patterns among the majority of solutions, although, none being very accurate visually.

Figure 4.79 illustrates the fittest solution out of the 30 runs. There is a semblance of a horizon line which then begins to gradient and dissolve into a curve. In terms of the behaviour comparison, the fittest solution sits in entropy 2.28-3.42 while the target is also in 2.28-3.42 and the luminosity is in 0-36 while the target is in 108-144. The entropy has successfully been reached to the same interval while the luminosity continues to be far from the target. The coefficient is much more condensed in

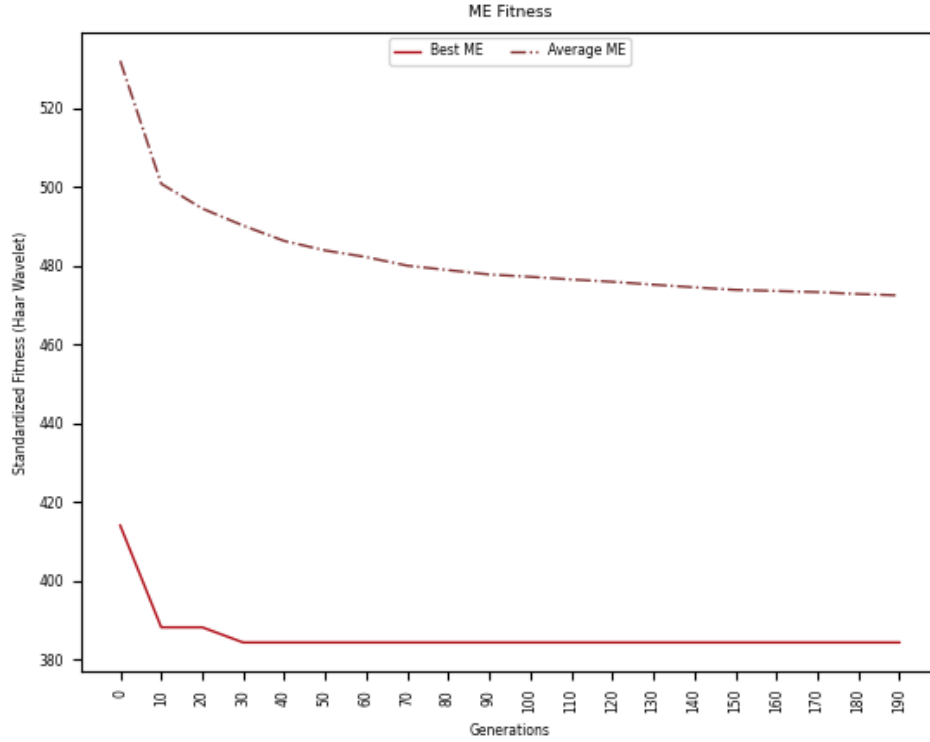


Figure 4.82: GP Fitness

comparison to the target's coefficients but hints at similar patterns.


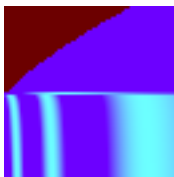
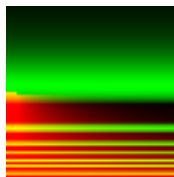


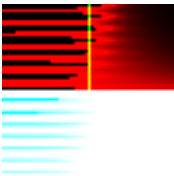

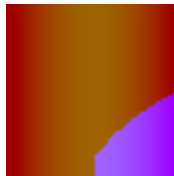


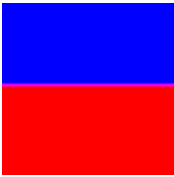
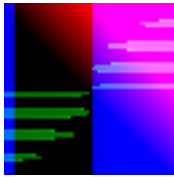
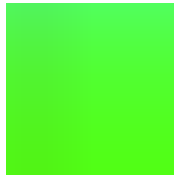

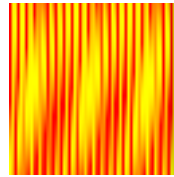
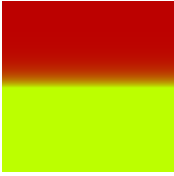
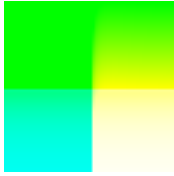
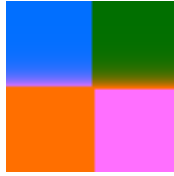
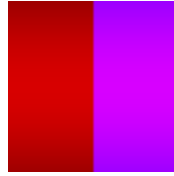
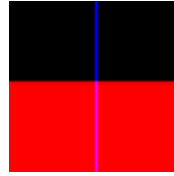



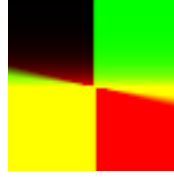


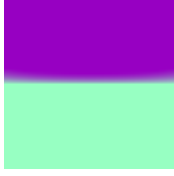
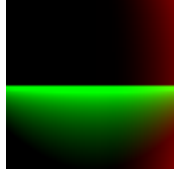


The MAP generated from this run in Figure 4.81 illustrates a visualization of the said MAP. There are very interesting traits between the luminosity and entropy cells. There are four general types of solution in each corner of the MAP with low luminosity and entropy, low luminosity but higher entropy, high luminosity and lower entropy, and high luminosity and entropy. Each of these categories contain certain geometries which slowly change as the cell values change.

The average fitness in Figure 4.82 shows a short but quick evolution. Both the average and best fitness improve quickly over 5 generations before slowly improving until generation 90 where it then improves very slightly for the rest of the generations almost converging. The best fitness converges very early, around the 30th generation. It improves sharply in the first 5 generations and slightly around generation 25-30.

4.3.6 MAP-Elite Results: SSIM

The MAP configurations remain the same as the previous two experiments in Table 4.13.

Table 4.25: ME Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

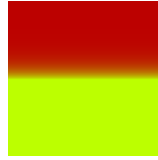


Figure 4.83: Best Solution

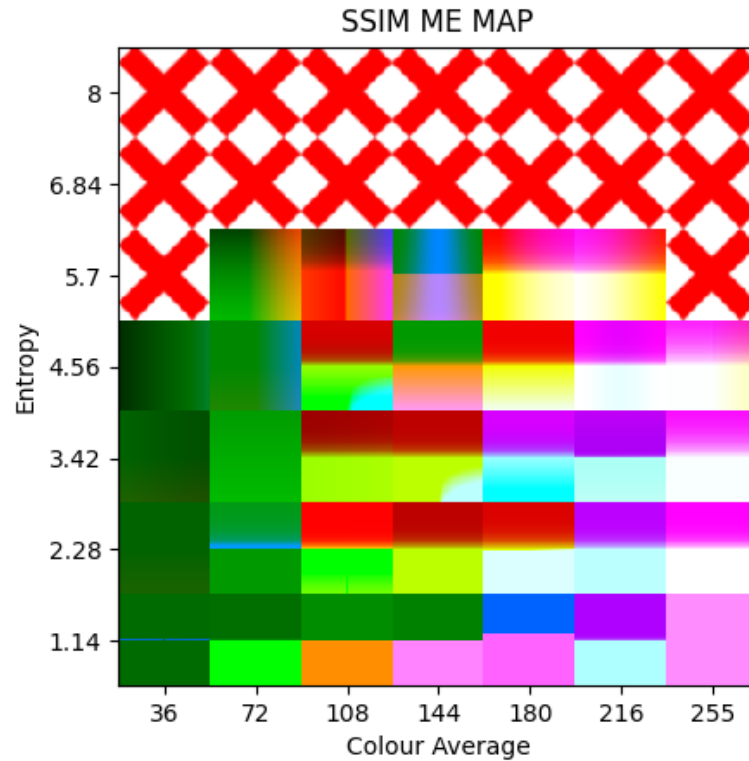


Figure 4.84: SSIM ME Visualization

The 30 runs from ME provide quite a diverse set of best solutions in Table 4.25. There are quite a few solutions opting in for the horizon line, however, with many more variations on it. For example, run 2, 7, 13, 17, and 18 modify the horizon line itself, have an interesting cone gradient on them, or have curious geometry on the image. Other images have taken a creative liberty on their own providing completely different designs like run 9, 14, and 27.

The best solution from the 30 runs in Figure 4.83 evolves a blurred horizon line. It has the behaviour intervals of entropy 1.14-2.28 and luminosity 108-144. The target

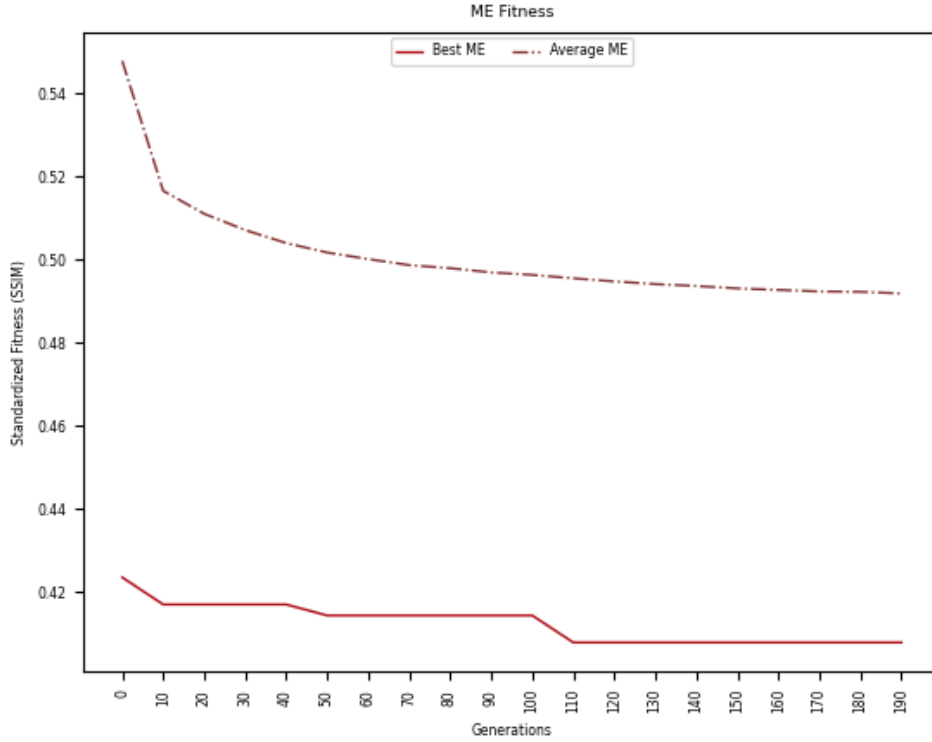


Figure 4.85: ME Fitness


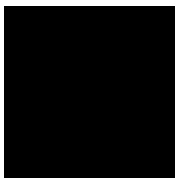

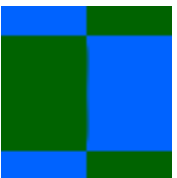
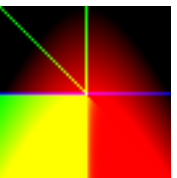

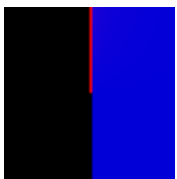
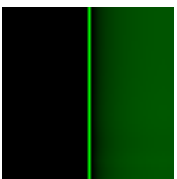

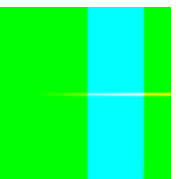

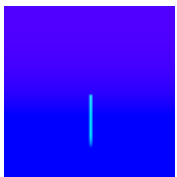
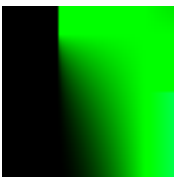
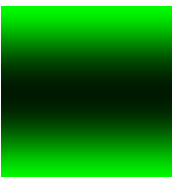
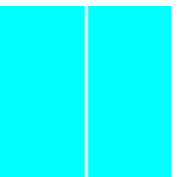
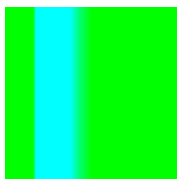
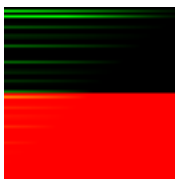
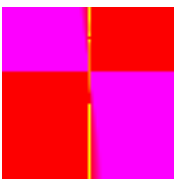
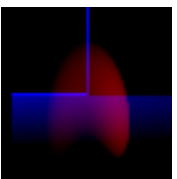

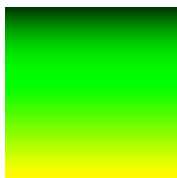

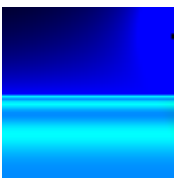
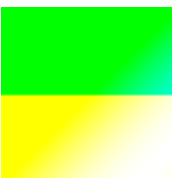
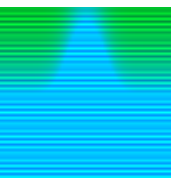
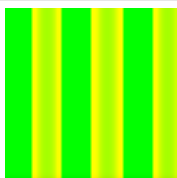
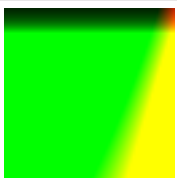
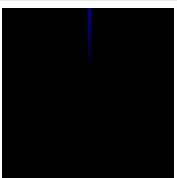

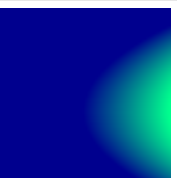
image resides in entropy 2.28-3.42 and luminosity 108-44. It successfully matched the luminosity interval but was off on the entropy. The resulting MAP is in Figure 4.84 where the majority of solutions follow the variations on the horizon line. There is one very interesting solution in the cell with colour average 144 and entropy 5.7 where it looks like the blue down the middle is an attempt at the sun.

The fitness quickly improves over the first 5 generation in Figure 4.85, however, from initialization it is drastic. Afterwards it continues to slowly improve for the next 100 generations before slowing down further. The best fitness slightly improves around generation 5, 40, and 110 before converging.

4.3.7 DME Results: Wavelet

This experiment utilizes the same configuration and behaviour set as Table 4.16.

Table 4.26: DME Image Results using Target Figure 4.63

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

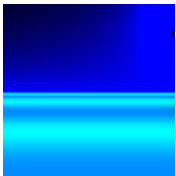
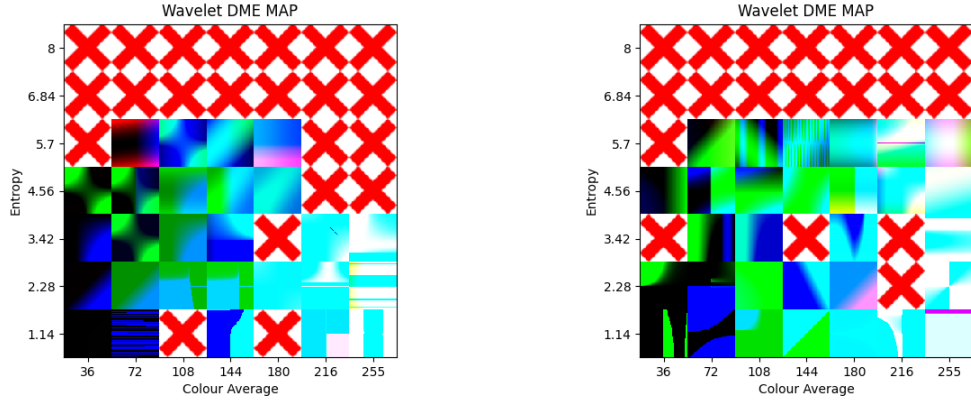


Figure 4.86: Best Solution

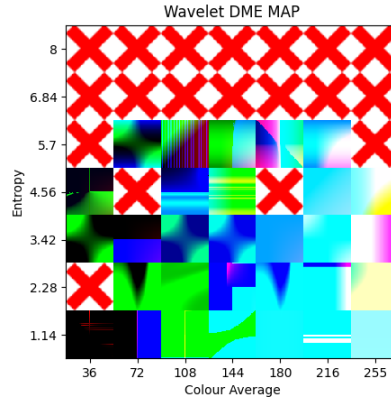


Figure 4.87: Best Solution Coefficient



(a) Entropy and Luminosity Island MAP 1

(b) Entropy and Luminosity Island MAP 2



(c) Entropy and Luminosity Island MAP 3

Figure 4.88: Wavelet DME Visualization

The best solutions in Table 4.26 produces a wide range of solutions. Some of them produce horizon lines, others opt for a vertical line, and some interesting geometry and patterns. Runs 22 and 29 standing out the most having horizon lines with geometry overlaying or protruding from the middle similar to that of a sun albeit stretch the description by quite a margin.

The best solution in Figure 4.86 showing a horizon line with some various gradients and stripes throughout. The behaviours giving an entropy of 3.42-4.56 and a luminosity of 72-108 overshooting the entropy by one interval and undershooting the luminosity by one as well. The coefficients are much more condensed than that of the

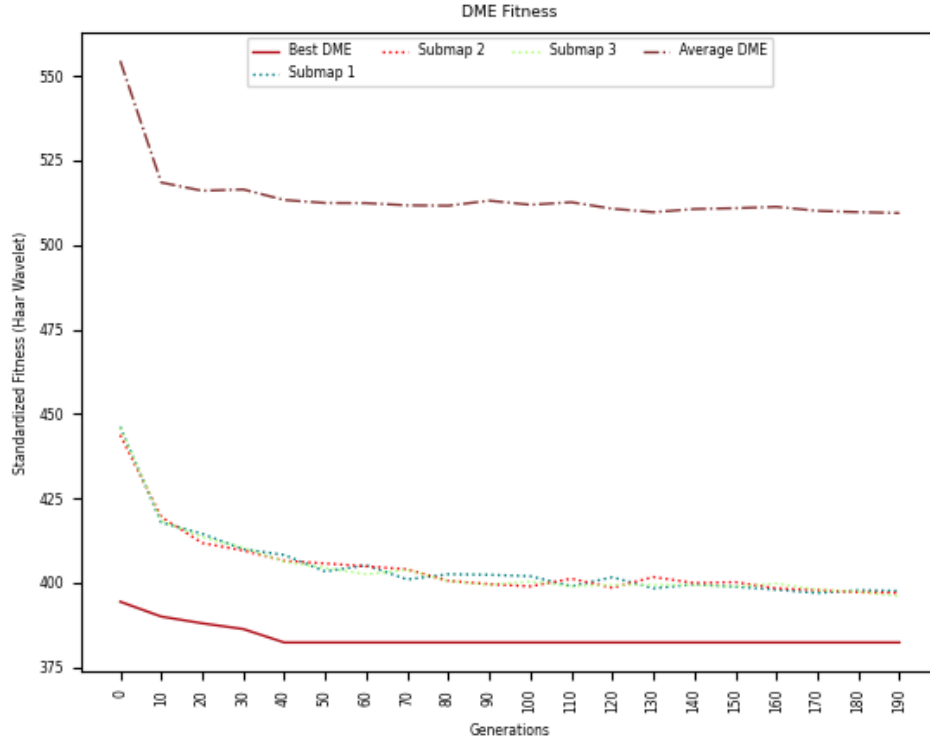


Figure 4.89: DME Fitness

target image although being in similar places along the top and left.

The sub-map configuration in Figure 4.88 shows other solutions which seem to be closer to the target than the best solution. There are numerous solutions spread throughout the configuration which show a horizon and an oval-like shape coming through it on both sides which indicates similarity to the sun and its shadow.

Figure 4.89 shows an improvement in submap fitness very quickly from generations 0 to 15 before slowing down slightly until generation 70. From generation 70 onward the fitness improvement is very slight before seemingly coming to a halt at generation 195. The best fitness only having marginal improvements until, roughly, the 40th generation before converging.

4.3.8 DME Results: SSIM

This experiment utilizes the same configuration and behaviour set as Table 4.16.

Table 4.27: DME Image Results




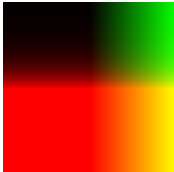



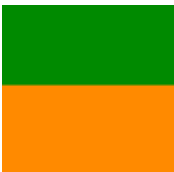
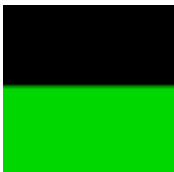



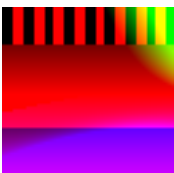


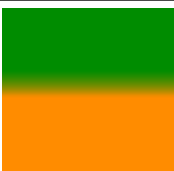
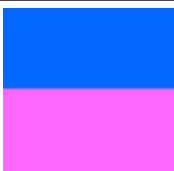
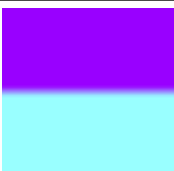
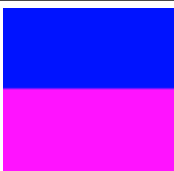



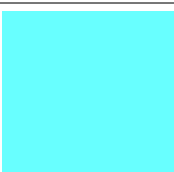
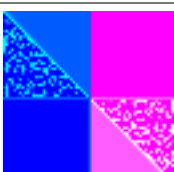
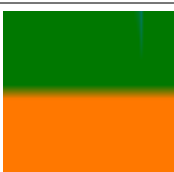


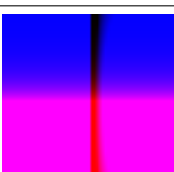
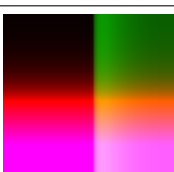
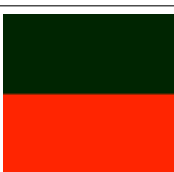
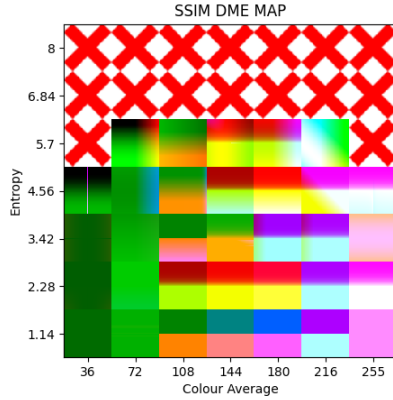
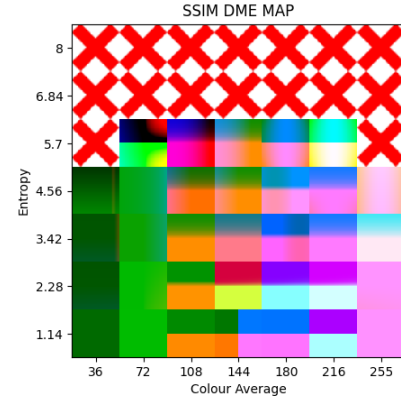
Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				



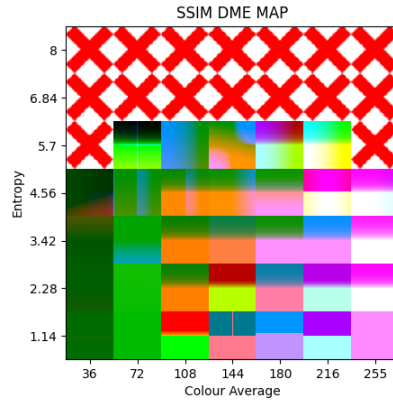
Figure 4.90: Best Solution



(a) Entropy and Luminosity Island MAP 1



(b) Entropy and Luminosity Island MAP 2



(c) Entropy and Luminosity Island MAP 3

Figure 4.91: SSIM DME Visualization

The 30 runs in Table 4.27 also show the majority of horizon lines with slight variations, however, a good number of runs seemed to favour the orange and green colours for the top and bottom. There are interesting details on the solutions for runs 15, 23, and 24.

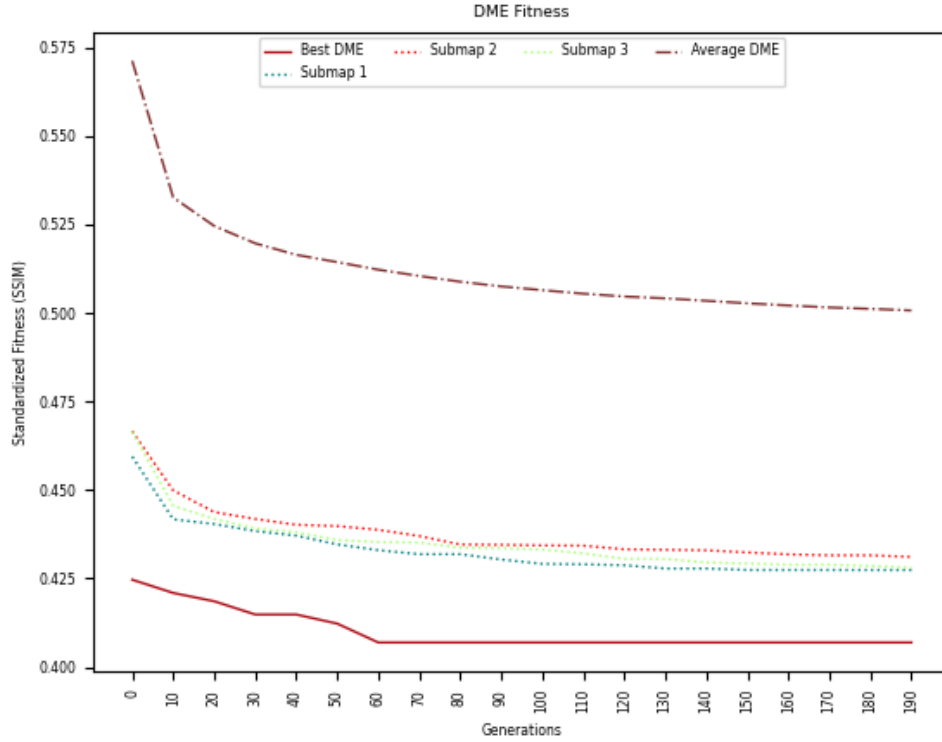


Figure 4.92: DME Fitness

The best solution of the 30 runs in Figure 4.90 does not opt in to the common green and orange split, but for a blue and purple with a flat horizon line. When comparing behaviours, the target has entropy 2.28-3.42 and luminosity 108-144 where this solution has entropy 0-1.14 and luminosity 108-144. The entropy was quite off while the luminosity interval is the same. The resulting MAP configuration from this run shows the common green orange solution split with a few other interesting outcomes which seem to occupy the higher entropy slots.

The fitness plot in Figure 4.92 highlights the three sub-maps evolving in a similar pattern with sub-map 3 performing the best out of all of them. Most of the evolution was done by generation 50 before slowing down. Sub-maps 1 and 2 had more fitness improvement until generation 175 meanwhile sub-map 3 meandered until the end. The average fitness in this case is a very smooth curve, unlike the others, making

large improvements in the first 30 generations before slowly improving until the end. The best fitness improves in two areas, the first 30 generations and from generations 50-65 before converging.

4.3.9 Discussion: Wavelet

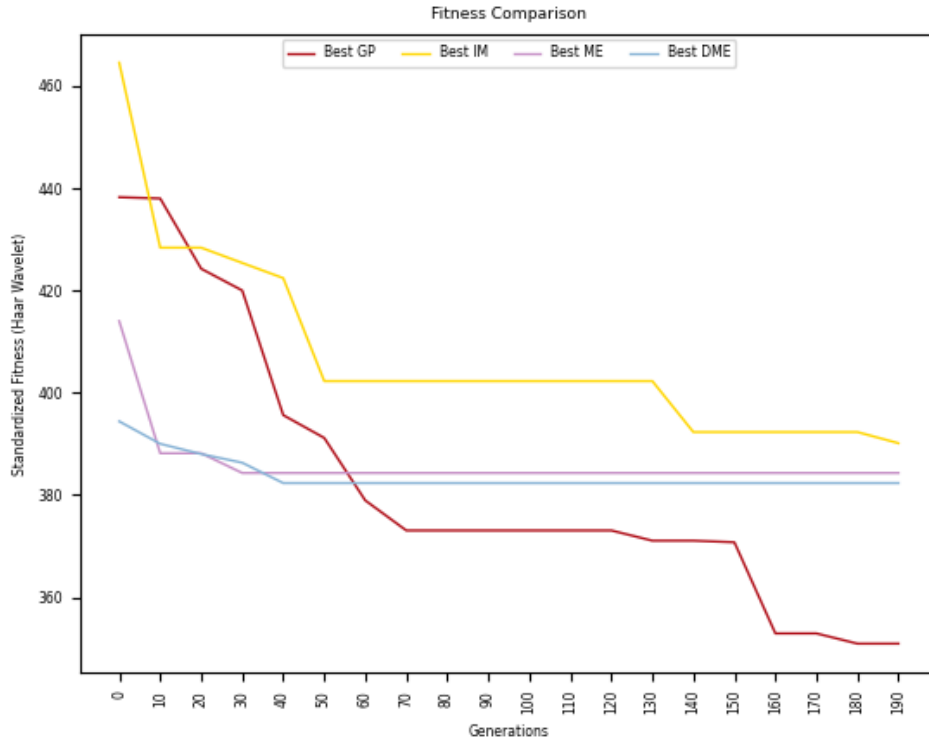


Figure 4.93: Wavelet Fitness Comparison

The fitness comparison in Figure 4.93 shows a large improvement for all the algorithms in the beginning. ME and DME coming to convergence the quickest, around generations 30 and 45 respectively. On the other hand, GP and IM slow down, but don't seemingly converge until much later. IM slows down around generation 50, and GP around generation 70. Both of them are mostly flat afterwards bar two areas for IM and one for GP. IM makes more improvements around generations 130 and 195. As for GP, it makes one last large improvement around 160, and another small one

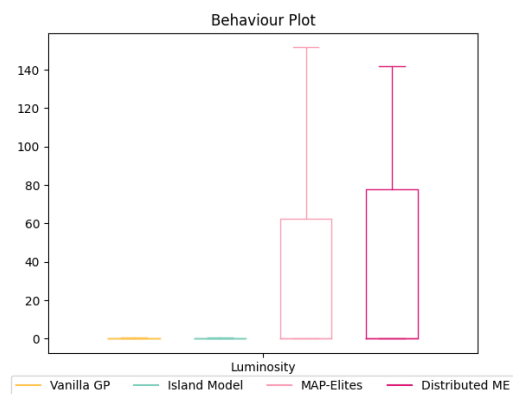


Figure 4.94: Luminosity Box Plot

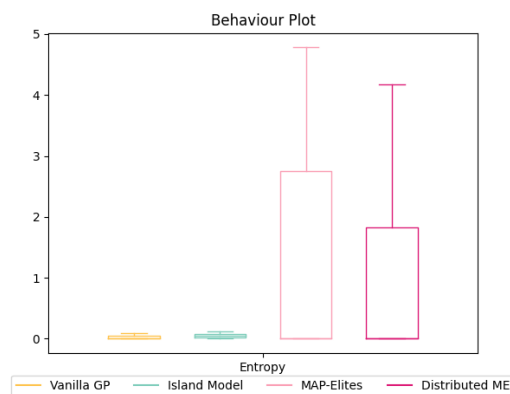


Figure 4.95: Entropy Box Plot

around generation 180.

The behaviour box plots in Figures 4.95 and 4.94 have very low medians for all algorithms in both entropy and luminosity. Vanilla GP and island-model do not have any runs within the 75th percentile which go further than the lowest values. Meanwhile ME and DME have very expansive interquartile ranges as well as very large whiskers.

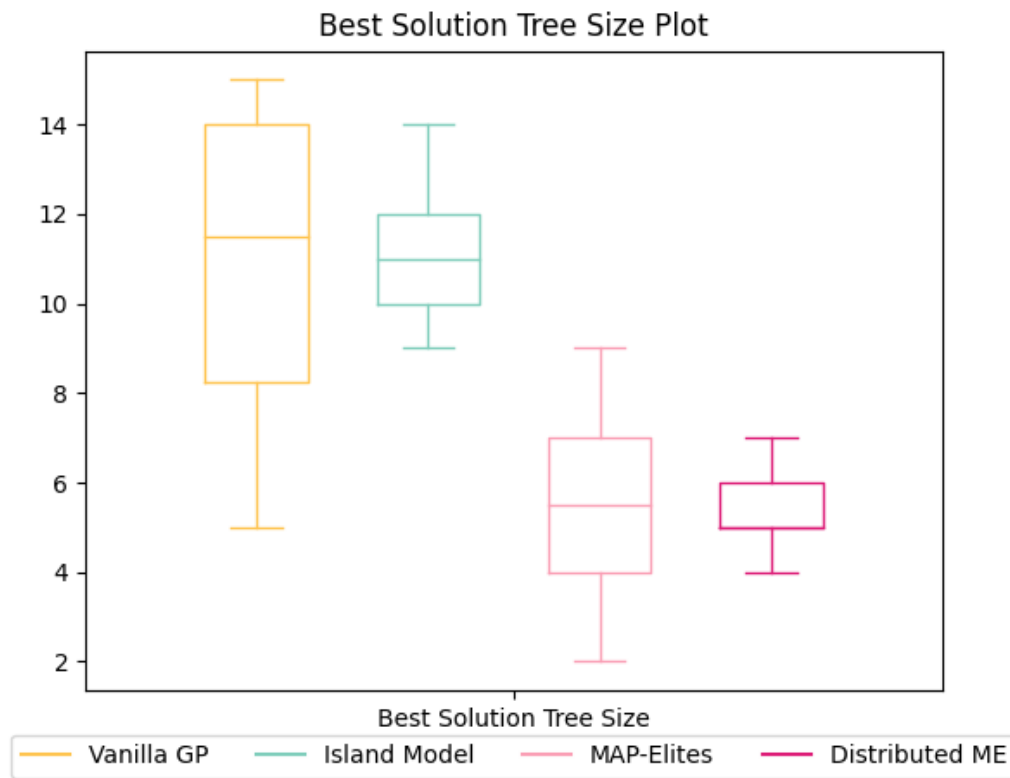


Figure 4.96: Tree Depth Plot

The tree plot in Figure 4.96 shows vanilla GP and island-model having close median depth of trees for the best solutions and the same for DME and ME. The single population variants have larger interquartile ranges as well as the whiskers going further than their multi-population counterparts.

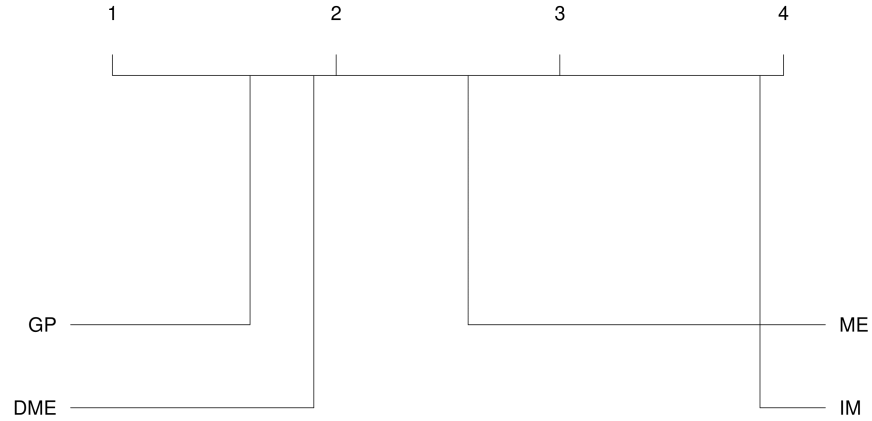


Figure 4.97: Critical Difference Diagram using best solutions from each algorithm

All following statistics have been formulated within R utilizing the Friedman test and Shaffer correction using a threshold of $p < 0.05$, with the null hypothesis:

$$H_0 : \theta_{GP} = \theta_{IM} = \theta_{ME} = \theta_{DME}$$

The four algorithms in Figure ?? are statistically significant due to the lack of black bar intersecting them. Therefore, there is enough evidence to state that the medians between the four algorithms are not the same. GP, DME, and ME are ranked higher than IM with GP ranked the highest, however, quite low and close to DME.

4.3.10 Discussion: SSIM

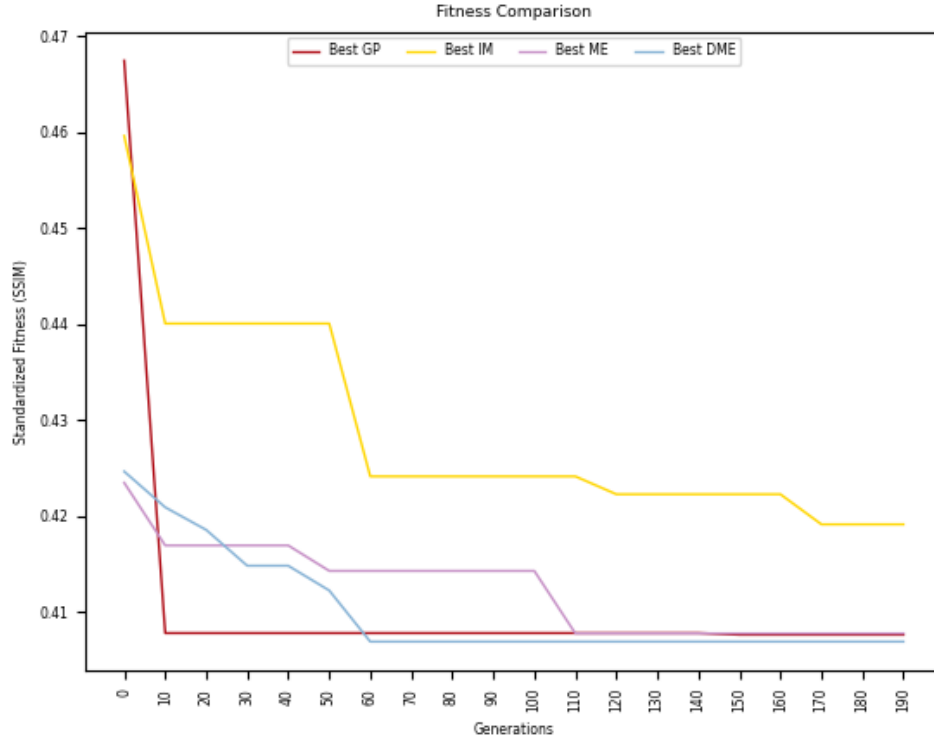


Figure 4.98: Wavelet Fitness Comparison

Figure 4.98 shows DME performing the best with GP and ME somewhat overlapping each other with IM lagging behind. GP quickly evolves and finds its best fitness within 15 generations. DME has a staircase like improvement for its best fitness until generation 60. ME improves in 3 spurts, around generation 6, 40, and 110 before being very close to GP. As for IM, it improves with great strides near generations 10 and 55 before sparsely making incremental improvements.

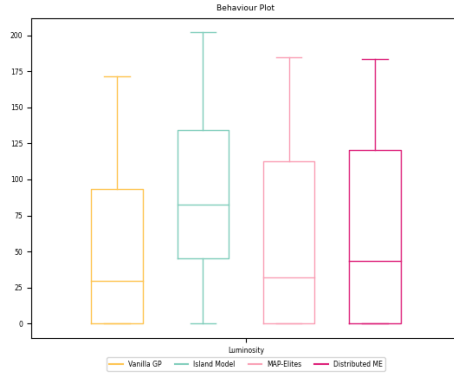


Figure 4.99: Luminosity Box Plot

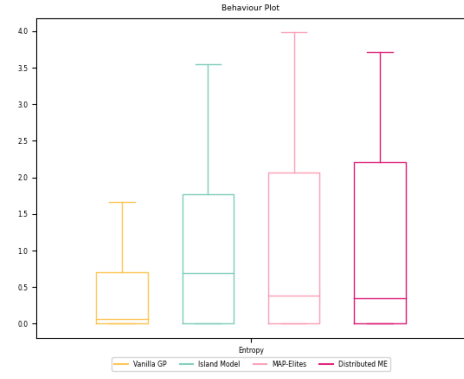


Figure 4.100: Entropy Box Plot

The behaviour box plots in Figures 4.99 and 4.100 are quite competitive overall. Figure 4.99 being the more competitive one with island-model having the highest median luminance which is closer to the target than that of vanilla GP and DME which have quite close medians. ME having the lowest median out of them all. The whiskers showing ME and DME having the highest maximums in the 75th percentile. The entropy behaviour plot in Figure 4.99 shows vanilla GP, ME, and DME with very low median entropies while island-model once again having the highest median. Vanilla GP and island-model's whisker lagging more than with luminosity in comparison to ME and DME who, once again, have the largest whiskers.

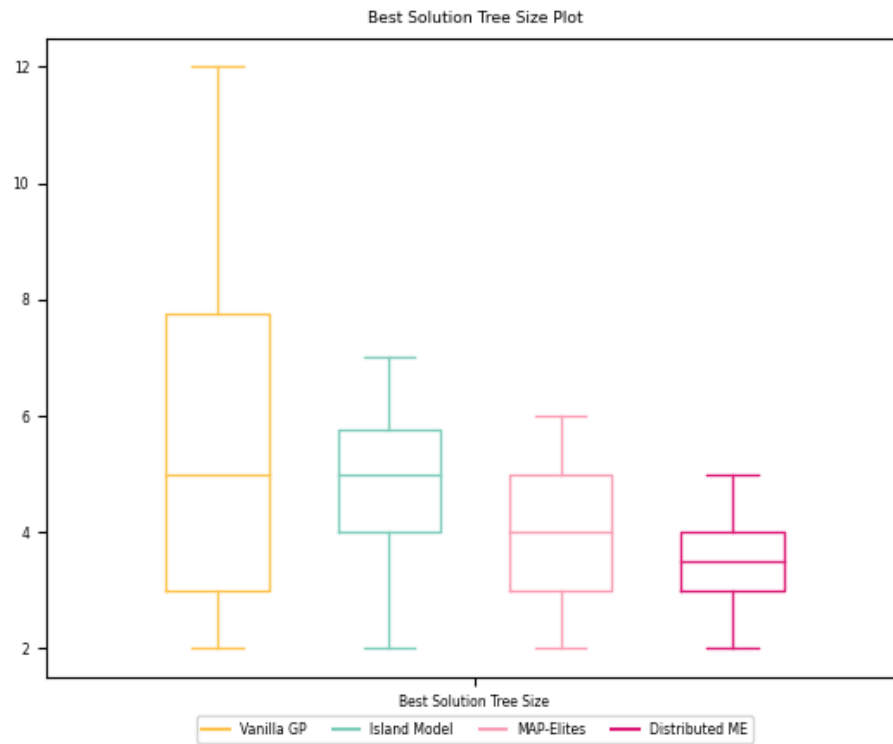


Figure 4.101: Tree Depth Plot

The tree depth plot in Figure 4.101 shows the median tree depths being quite close to one another. ME and DME producing the smallest trees having the smallest interquartile range and whisker, however, island-model is also creating generally small trees especially in comparison to vanilla GP.

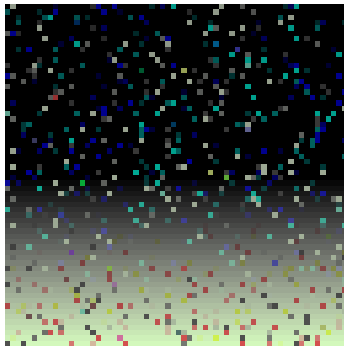
fitness functions are required. The behaviours used in this experiment will be the same once again as in Table 4.8.

Fitness Function 1	Haar Wavelet Coefficient Comparison.	Ranking x highest value coefficients and comparing the respective indice's rank.
Fitness Function 2	SSIM	Calculating the SSIM over every window in the image.
Fitness Function 3	CHISTQ	Quantize the colours and calculate distance and similarity.

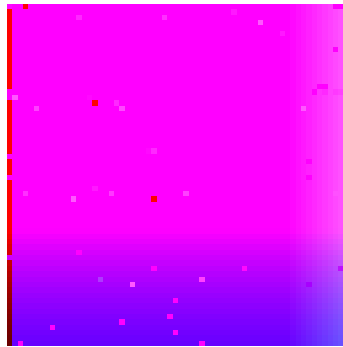
Table 4.28: Fitness Function and Behaviours

4.4.1 Island-Model Results

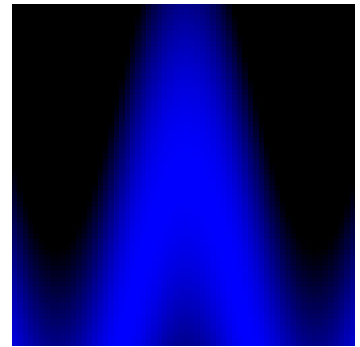
In Appendix A Section A.1 there are the best solutions from each subpopulation in Tables A.1, A.2, and A.3. Since each of the islands utilize a different fitness function with their own scales as well as calculating fitness in completely different ways they will not be put into a direct table comparing one another.



(a) CHISTQ Best



(b) SSIM Best



(c) Wavelet Best

Figure 4.103: Best Subpopulation Solutions

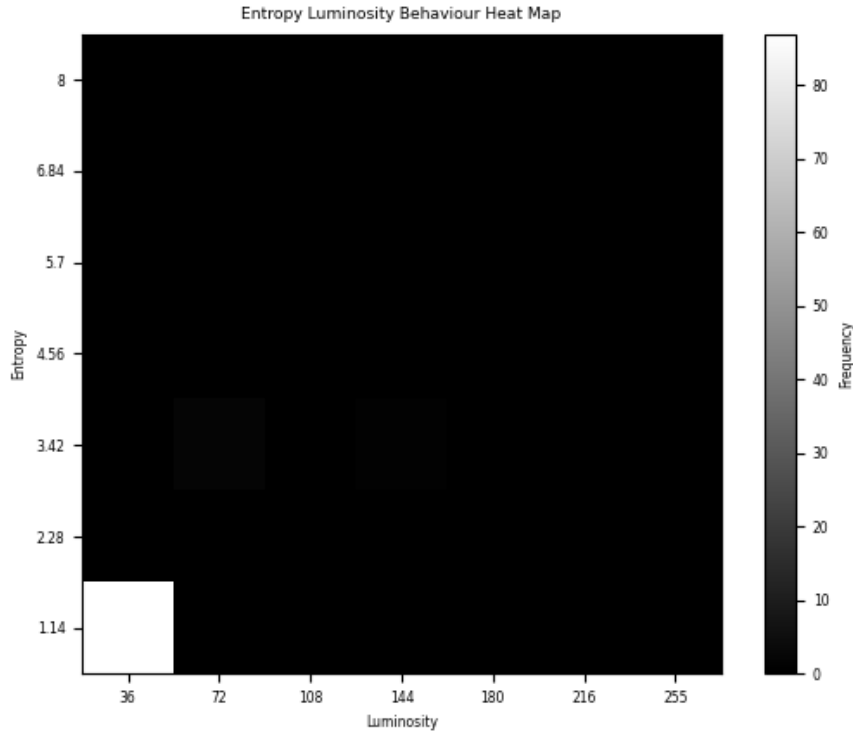


Figure 4.104: Entropy Luminosity Heat Map

Figure 4.103 shows the best solution using each fitness out of the 30 runs. The first subpopulation utilized the CHISTQ fitness, the second using SSIM, and the third using wavelet. SSIM and CHISTQ seem to both have evolved gradients in their respective runs while wavelet came up with a triangle protruding from the middle. The two gradients also have pixels littered throughout the image, CHISTQ more densely so than SSIM. When comparing behaviour intervals, Van Gogh resides in entropy 4.56-5.70 and luminosity 108-144. In comparison, the best from CHISTQ is in 2.28-3.42 and 36-72, SSIM's best is in 1.14-2.28 and 108-144, and wavelet's best is in 1.14-2.28 and 108-144. Out of the three, CHISTQ's best entropy interval is the closest to the target image, however, wavelet and SSIM's best produce a matching luminosity interval.

The diversity heat map in Figure 4.104 shows the lack of diversity throughout by

having the solutions all with an entropy below 1.14 and a luminosity below 36.

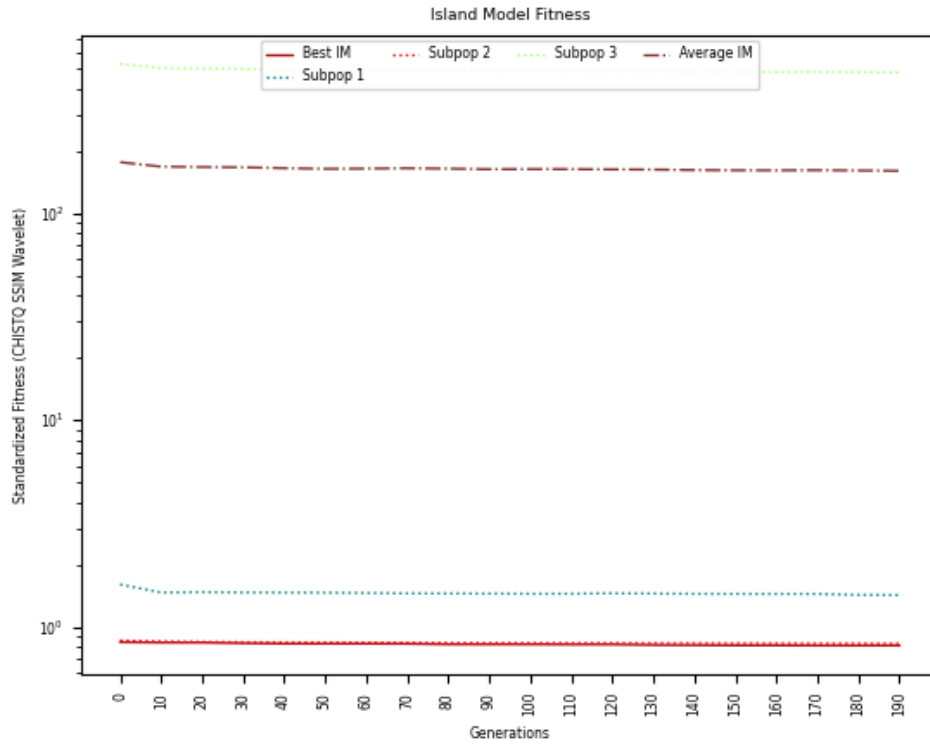


Figure 4.105: IM Fitness

The fitness plot in Figure 4.105 is log scaled due to the multiple fitness values being on very different scales. CHISTQ and SSIM are on much smaller scales where wavelet can easily go into the hundreds or even thousands. The subpopulations are seen improving slightly in the beginning before coming to a seeming convergence.

4.4.2 DME Results

This experiment utilizes the same configuration and behaviour set as Table 4.16.

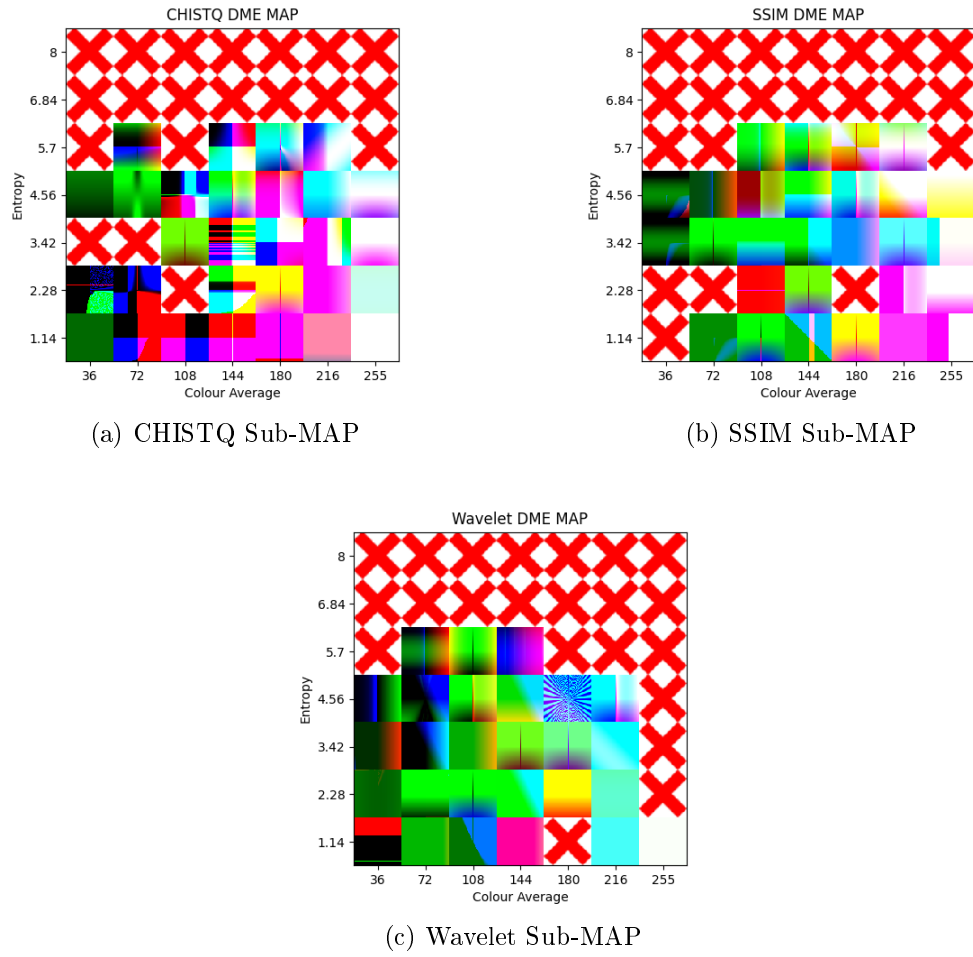


Figure 4.107: Best Overall Sub-MAP configuration

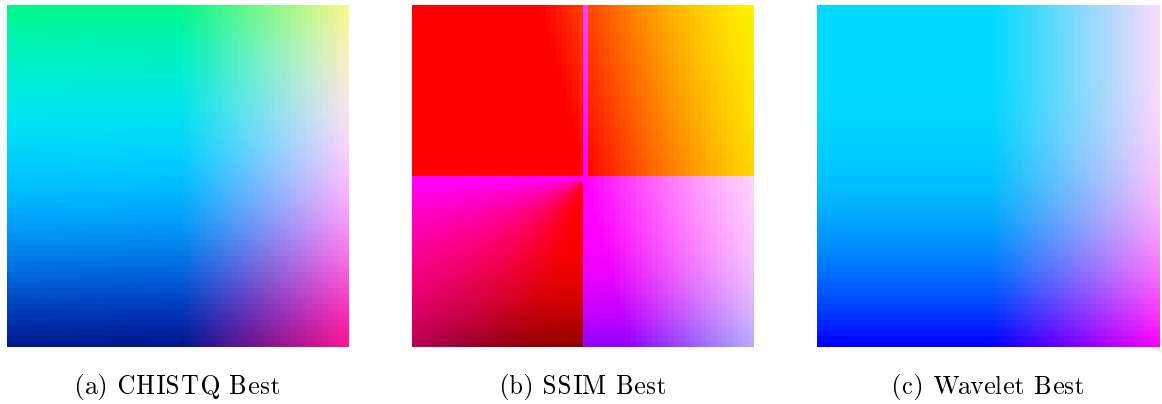


Figure 4.106: Best Subpopulation Solutions

In the best solutions from each sub-map in Figure 4.106 it seems they all have the

same general archetype of multiple colour gradients. SSIM takes a slight change of direction by also splitting the image into 4 squares. Van Gogh's behaviour intervals are entropy 4.56-5.70 and luminosity 108-144. When comparing DME's best to it CHISTQ has an entropy of 4.56-5.70 and luminosity 144-180, SSIM with an entropy of 3.42-4.56 and luminosity 108-144, and wavelet with an entropy of 3.42-4.56 and luminosity of 144-180. CHISTQ manages to find a solution that matches the entropy interval of the target, while SSIM and wavelet are close. SSIM manages to match the interval for luminosity where CHISTQ and wavelet are also one interval off.

Figure 4.107 represents the MAP configuration when calculating the average fitness over the 3 sub-MAPs. There are many solutions throughout with lines down the middle which can be attributed to the frequent immigration and emigration of individuals. Otherwise, CHISTQ's sub-MAP has the most interesting geometry with wavelet then producing the next most visually intriguing solutions while SSIM's solutions tend to look quite plain.

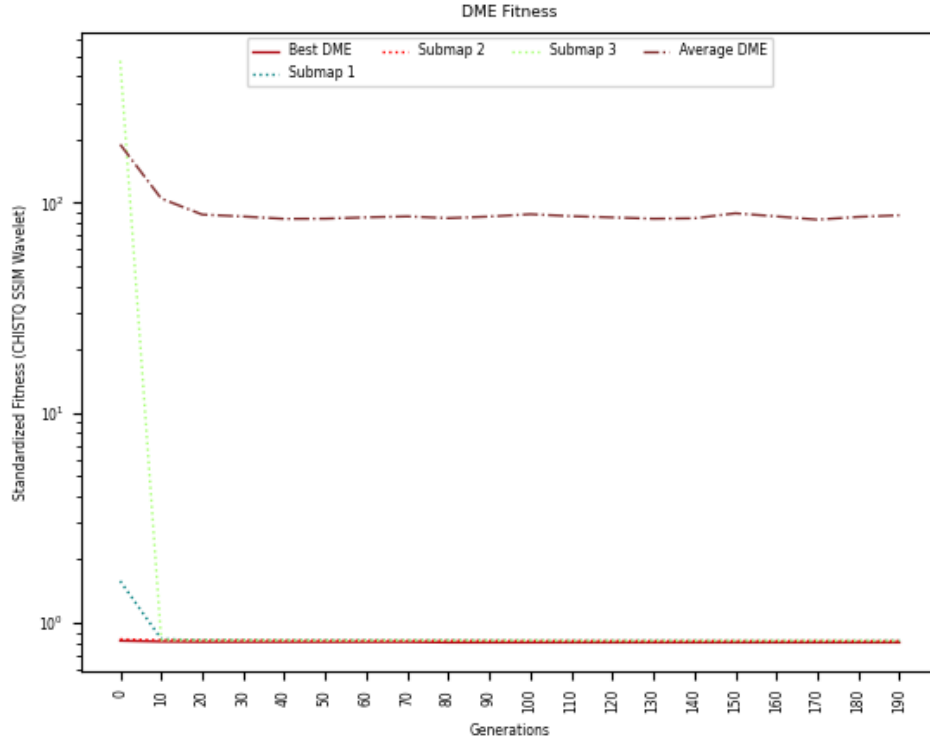


Figure 4.108: DME Fitness

The fitness plot in Figure 4.108 is log scaled like in IM in for the same scaling reasons. shows a very quick improvement within 5 generations in the wavelet submap. The CHISTQ submap also makes a large improvement within the first 10 generations.

4.5 Discussion

The fitness comparison in Figure 4.109 averages the three different fitness values for comparison to provide a more encompassing view on the data. This is because each subpopulation/submap uses its own fitness functions which work differently. DME far outperforms IM due to the difference that in wavelet fitness.

The behaviour plots in Figure 4.110 indicate a similar luminosity throughout the runs. Island-model tended to evolve with a higher luminosity than DME with the

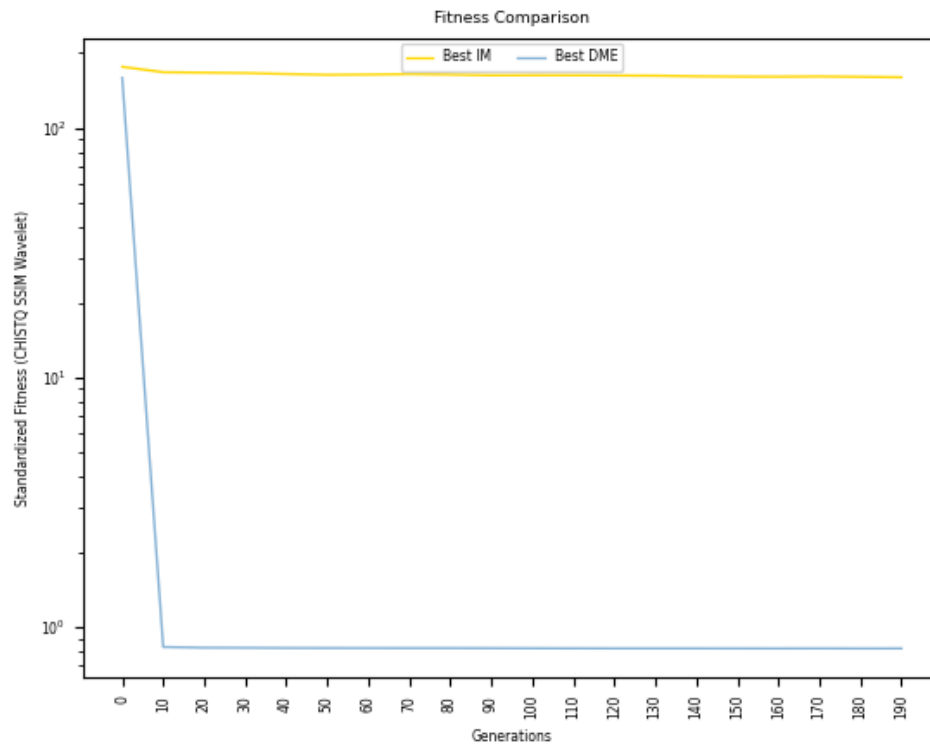


Figure 4.109: Fitness Comparison

median line hovering closer to 75 and DME around 50. Entropy paints a slightly different story with the interquartile range of DME being much larger than that of island-model. However the maximum of the third quartiles are quite close to one another as well as the medians.

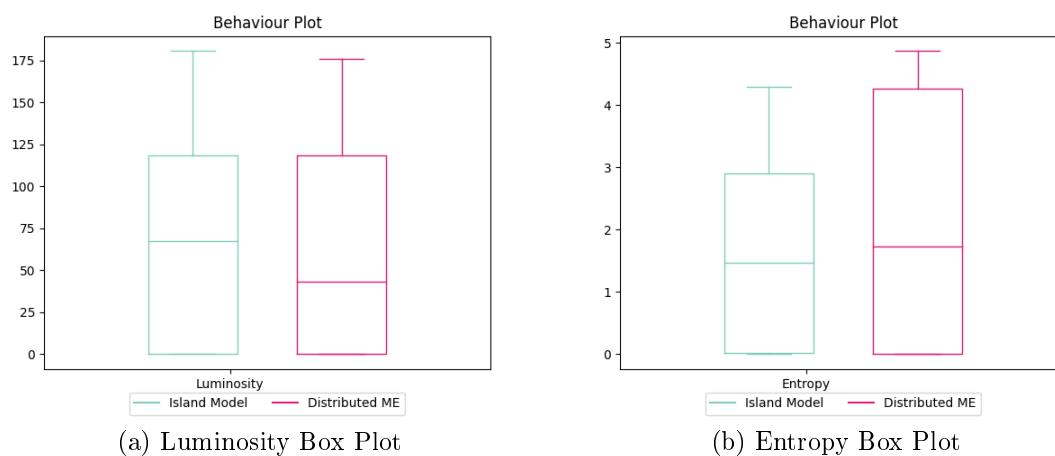


Figure 4.110: Entropy and Luminosity Behaviour Box Plots

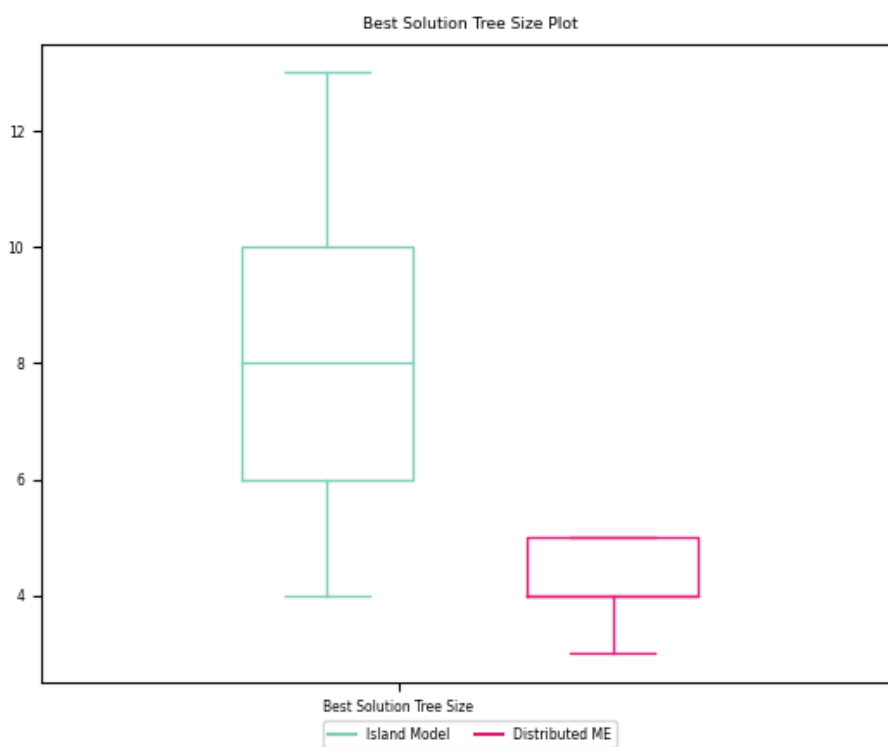


Figure 4.111: Tree Depth Box Plot

Figure 4.111 illustrates the difference in tree depth between island-model and DME. The maximum of the third quartile is equivalent to the upper quartile of the

interquartile range with the median being equivalent to the lower quartile of the interquartile range. The median of DME is very close to that of the lowest value of the first quartile of island-model where island-model's median is close to double of DME.

4.6 Comments

The primary focus of algorithm assessment is the quality and diversity. Quality is strictly in terms of the fitness comparison. Diversity is in terms of the number of solutions produced at the end of the run.

IM had the best end-of-run fitness in the direct match experiment, GP performed the best in the barcode wavelet experiment and the sunset wavelet experiment, ME outperformed in the wavelet cartoon face, and DME outperformed in SSIM and CHISTQ SSIM wavelet experiment.

Generally speaking, the fitness values are quite competitive, disregarding the CHISTQ SSIM wavelet experiment. Albeit, the differences are statistically significant from one another, bar the direct match experiment where the ME variants are only significantly different than GP and IM.

Diversity is more clear cut than the quality of solutions. GP produces one solution at the end of the run. IM produces as many solutions as there are islands, in the case of this research, 3 solutions. ME produces at most 343 solutions in the direct match experiment, and at most 49 in all other experiments. DME produces at most 147 individuals at the end of the run.

GP being far behind the rest when it comes to diversity. IM can provide some diversity but cannot be compared to the ME variants. ME depends on the number of behaviours used, however, still a large amount of individuals. DME produces less individuals than ME when the behaviours scale, however, when comparing equivalent

behaviours DME is more diverse than ME.

When comparing ME to DME the visualization capability is proved to be very nice in comparison to an image dump into a folder for high dimensional ME. Having the MAPs be generated through post-processing saves time and provides an easily distributable and understandable format when dealing with the results of a run. When DME and ME are both 2 dimensions resulting in two cases of visualization, DME is to ME what is island-model is to vanilla GP.

The textures evolved are very poor when observed visually in comparison to the target images. This can be attributed to the extremely basic GP language which only utilizes simple mathematic functions. Complex math functions as well as geometric functions would be very beneficial but at a high computational cost.

DME's visualization and islands were helpful when observing solutions. On top of this, when using a basic 2 behaviour setup the amount of solutions at an arms reach provided many more interesting solutions in comparison to the same 2 behaviour setup for ME.

DME, on average, produced the smallest trees bar the first experiment. Even when it didn't produce the smallest trees, it was very close behind ME, the opposite is true for ME when DME would produce smaller trees than it. Smaller trees result in faster run time due to the need to evaluate less than a larger tree. There is also the argument that a larger tree may not produce better solutions due to bloating and sub-trees not producing anything useful.

DME still suffers from the same problems that island-model suffers which is parameter optimization. ME already has many specific parameters to itself, and on top of that DME requires the same island parameters which are in island-model. This results in numerous parameters which can be optimized for. For the sake of time constraints, parameter optimization was not considered, and a simple GP language was used due to the amount of time it would take to run a complex language. Similarly

to island-model there is only a possibility of super linear speed-up due to the nature of being problem specific.

Chapter 5

Conclusion

The goals of this thesis are: to use multiple behaviours without exponential increase in the total potential individuals; to compare the diversity and quality of solutions of MAP-Elites to the baseline runs; discover a practical strategy to examine multiple solutions; and to test DME on an evolutionary design problem. The proposed DME system successfully used sets of 2 and 3 behaviours with only linear growth in total potential solutions when used on procedural texture problems. While MAP-Elites provides an exponential growth of total potential solutions, this linear growth of DME greater reduction of potential solutions compared to the MAP-Elites hypercube representation of total potential solutions. MAP-Elites was compared to the baselines of vanilla GP and island-model in every experiment, showing a much more diverse solutions with competitive, and in some cases, better fitness than the baseline runs. As seen in the results of the experiments, the DME MAP visualization was successful in all cases.

The first experiment shows the concept of using high-dimensional MAP-Elites and a comparison between the four algorithms to highlight the usage of DME as proposed. The second and third experiments test simple and difficult target images to see how useful ME and DME are when both can be visualized into a plot. The second uses the same fitness function for both target images, and the third experiment tries a more modern fitness function. The fourth experiment expands on this, utilizing different fitness functions for each island and drawing a comparison between the island-model evolution and DME algorithm. Throughout all experiments, it is seen that the most fit individual produced is not necessarily the most visually accurate solution which has been produced. The solution diversity of ME and DME algorithms is greater than vanilla GP and island-model evaluation. Since Experiment 1 could not be visualized only DME was proven useful in finding interesting solutions or solutions attempting

to be close to the target given the fitness function being very strict. In the second and third experiment this was proven the most given simpler target images than the first and fourth experiment. DME and ME found solutions which were very close to being visually spot on or finding key aspects that were seen in the target. The first and fourth experiment use the same target image and struggle to evolve good solutions through visual observation.

5.1 Issues

5.1.1 High Level Issues

- GP parameter optimization
- Island parameter optimization
- ME parameter optimization

5.1.2 Low Level Issues

- GP language
- Wavelet algorithm
- SSIM algorithm
- CHISTQ algorithm
- Java heap space
- Java speed and threading
- MAP functions implementation

GP inherently has many parameters to work with, and thus, a notable amount of parameters to select for efficient performance. Island-model evolution introduces more parameters to the already existing GP parameters through migration and the islands each being their own instance of typical GP evolution. MAP-Elites is a different kind of evolution and replaces some of GP's parameters, but at the same time introduces more MAP-Elite specific parameters. Therefore, DME combining MAP-Elites and island-model evolution has a slurry of parameters to work with, making it even more

difficult to choose from. Note that, there is research exploring parameter performance prediction which would help with these problems [35].

The GP language used for this research is considered a basic procedural texture language. There are many geometric functions and noise generators which can be added to the language and have shown to provide better results, fitness-wise and visually. This basic language resulted in visually unimpressive solutions, but they were efficient to compute.

As mentioned earlier, the wavelet fitness uses a customized truncation for the highest magnitudes. This implementation was not thoroughly investigated or explored, which introduces some questions with the performance of this fitness. SSIM was proven to perform better with a different window technique rather than the sliding one used in this thesis [53]. The other window technique would have provided different, possibly better, solutions than the one used in this research. As for CHISTQ, this is a decades old image comparison algorithm. Newer algorithms should be explored rather than CHISTQ.

The implementation of the system in Java introduced some heap space issues when implementing the individuals. The speed and threading of Java could be further explored and improved.

Although island-model evolution, and by in part, Distributed MAP-Elites, have properties of linear separability, this research does not confirm a general applicability of the system to such problems that are not linearly separable. The evolutionary design application was chosen without respect to problems being linearly separable. With respect to fitness and optimization, if the problem is not linearly separable then the diversity produced by DME will provide no aid to the optimization through reproduction. However, the islands migrating individuals and exploring the search space may have some effect on the evolution, though this would need to be thoroughly researched. Although, it is to be noted that GP is capable of solving non-linearly separable problems [27].

The system introduces a diverse set of solutions which each take up a spot within the search space as well as the capability to visualize these solutions if possible. If the problem is not capable of producing solutions that can be visualized, it still cuts down on the total candidate solutions, and maintains the unique relationship pairs of any number of behaviours being measured. The end-of-run analysis of solutions will still be easier than MAP-Elites, due to the cut in total candidate solutions, on top of the capability of exploring the relationships between behaviours and the behaviours to the fitness.

5.2 Future Work

On an implementation level the system can be improved on in terms of its memory and speed. With procedural textures on this system, GPU acceleration can be looked into for a many-times speed-up or general speed-up for evolutionary design problems.

Within the first experiment, an arbitrary usage of luminosity direct match was used in conjunction with behaviours of mean red, green, and blue values which do not have a meaningful correlation with one another. Alternatively geometric behaviours while evolving the solutions with a colour based fitness would be much more valuable.

The second experiment uses a custom evaluation on the Haar wavelet transformations to provide a fitness value. Since it utilizes the number of highest-ranked coefficients within the image the number of coefficients used is a completely separate problem to optimize for. The penalty involvement for the lack of finding a high ranked coefficient in the same pixel position also provides another layer of problems. The GP language utilized was a very simple language which affects the quality of solutions where the usage of a more complex language would produce better procedural textures.

The third experiment involves the usage of a third party program to evaluate SSIM fitness. It utilizes the 8x8 sliding window implementation which is the most basic. It has been identified that there are alternative window configurations which produce better results and are less rigid. However, like with the second experiment the simplified language is another area of contention where complexity may prove useful. The same things about the Haar wavelet transformation fitness can be said within this experiment.

The fourth experiment involved problems of the second and third experiment along CHISTQ. CHISTQ is an extremely outdated evaluation of colour distance through quantization utilizing similarity. Being decades old there are other newer approaches which are improvements. The GP language, and parameter optimization of the island-model parameters are things that can be optimized as well.

All these experiments utilized the same set of behaviours and interval counts, these can be explored to varying degrees to explore the relationships of different aspects of procedural textures which can further spur on more research.

Furthermore, the experiments conducted to test the DME system have been limited due to time. Many different aspects of evolutionary design can be explored using this system than just procedural textures which brings a plethora of new applications to consider. Applications specifically dealing with being able to visualize the solutions

like 3-D models, blueprints, design, etc. would benefit greatly from DME because of the capability of visualizing many solutions at the same time. Engineering problems where someone must consider the relationship between many different phenotypes or behaviours would benefit much more from ME where a single solution can show the relationship of many behaviours at a certain interval for each dimension.

More ambitious projects can also be branched off from this system like a interactive DME system where the users could choose elites to reproduce with one another. There is also the use of deep learning within the fitness function, or using DME as the input to a deep learning network.

Bibliography

- [1] Shakhnaz Akhmedova, Vladimir Stanovov, and Eugene Semenkin. Soft island model for population-based optimization algorithms. In Ying Tan, Yuhui Shi, and Qirong Tang, editors, *Advances in Swarm Intelligence*, pages 68–77, Cham, June 2018. Springer International Publishing.
- [2] Alberto Alvarez, Steve Dahlskog, Jose Font, and Julian Togelius. Empowering quality diversity in dungeon design with interactive constrained map-elites. In *2019 IEEE Conference on Games (CoG)*, pages 1–8, August 2019.
- [3] David Andre and John R. Koza. A parallel implementation of genetic programming that achieves super-linear performance. *Inf. Sci.*, 106(3–4):201–218, may 1998.
- [4] Illya Bakurov, Marco Buzzelli, Mauro Castelli, Raimondo Schettini, and Leonardo Vanneschi. Genetic programming for structural similarity design at multiple spatial scales. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '22*, page 911–919, New York, NY, USA, 2022. Association for Computing Machinery.
- [5] Sheikh Faishal Basher. Managing diversity and many objectives in evolutionary design. Master’s thesis, Brock University, 2021.
- [6] Dominique Brunet, Edward R. Vrscay, and Zhou Wang. On the mathematical properties of the structural similarity index. *IEEE Transactions on Image Processing*, 21(4):1488–1499, 2012.
- [7] Erick Cantú-Paz et al. A survey of parallel genetic algorithms. *Calculateurs paralleles, reseaux et systems repartis*, 10(2):141–171, 1998.
- [8] Erick Cantú-Paz. A survey of parallel genetic algorithms. *Calculateurs paralleles, reseaux et systems repartis*, 10:141–171, 1998.

- [9] Yisong Chen and Horace H.S. Ip. Texture evolution: 3d texture synthesis from single 2d growable texture pattern. *The Visual Computer*, 20(10):650–664, December 2004.
- [10] Cédric Colas, Vashisht Madhavan, Joost Huizinga, and Jeff Clune. Scaling map-elites to deep neuroevolution. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, GECCO '20, pages 67–75, New York, NY, USA, June 2020. Association for Computing Machinery.
- [11] F. Corno, E. Sanchez, and G. Squillero. Exploiting co-evolution and a modified island model to climb the core war hill. In *Proceedings of the 2003 Congress on Evolutionary Computation, 2003. CEC '03.*, volume 3, pages 2217–2221 Vol.3, 2003.
- [12] Joaquín Derrac, Salvador García, Daniel Molina, and Francisco Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.*, 1:3–18, 2011.
- [13] Emily Dolson, Alexander Lalejini, and Charles Ofria. *Exploring Genetic Programming Systems with MAP-Elites*, pages 1–16. Springer International Publishing, Cham, January 2019.
- [14] Grasielle Duarte, Afonso Lemonge, and Leonardo Goliatt. A dynamic migration policy to the island model. In *Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 1135–1142, 2017.
- [15] D.S. Ebert, F.K. Musgrave, D. Peachey, K. Perlin, and S. Worley. *Texturing & Modeling: A Procedural Approach*. Morgan Kaufmann Series in Computer Graphics and Geometric Modeling. Morgan Kaufmann, 2003.
- [16] Manon Flageat and Antoine Cully. Fast and Stable MAP-Elites in Noisy Domains Using Deep Grids. *ALIFE 2020: The 2020 Conference on Artificial Life*, pages 273–282, July 2020.
- [17] Matthew C. Fontaine, Scott Lee, Lisa B. Soros, Fernando de Mesentier Silva, Julian Togelius, and Amy K. Hoover. Mapping hearthstone deck spaces through map-elites with sliding boundaries. In *Proceedings of The Genetic and Evolutionary Computation Conference*, GECCO '19, pages 161–169, New York, NY, USA, July 2019. Association for Computing Machinery.

- [18] Michael Gircys and Brian Ross. Image evolution using 2d power spectra. *Complexity*, 2019, January 2019.
- [19] Yiyuan Gong and Alex Fukunaga. Distributed island-model genetic algorithms using heterogeneous parameter settings. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, pages 820–827, 2011.
- [20] Adam Hewgill and Brian Ross. Procedural 3d texture synthesis using genetic programming. *Computer and Graphics Journal*, 28(4):569–584, January 2004.
- [21] Alaa E. M. Ibrahim. *GenShade: An Evolutionary Approach to Automatic and Interactive Procedural Texture Generation*. PhD thesis, December 1998.
- [22] Alaa Eldin M. Ibrahim. Multiple-process procedural texture. *The Visual Computer*, 33(12):1511–1528, December 2017.
- [23] Dario Izzo, Marek Rucinski, and Christos Ampatzis. Parallel global optimisation meta-heuristics using an asynchronous island-model. In *Proceedings of the 2009 IEEE Congress on Evolutionary Computation*, pages 2301–2308, 2009.
- [24] Charles E. Jacobs, Adam Finkelstein, and David H. Salesin. Fast multiresolution image querying. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, page 277–286, New York, NY, USA, 1995. Association for Computing Machinery.
- [25] Niels Justesen, Sebastian Risi, and Jean-Baptiste Mouret. Map-elites for noisy domains by adaptive sampling. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '19, pages 121–122, New York, NY, USA, July 2019. Association for Computing Machinery.
- [26] Ahmed Khalifa, Scott Lee, Andy Nealen, and Julian Togelius. Talakat: Bullet hell generation through constrained map-elites. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 1047–1054, New York, NY, USA, July 2018. Association for Computing Machinery.
- [27] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.
- [28] B D H Latter. The island model of population differentiation: a general solution. *Genetics*, 73(1):147–157, 01 1973.

- [29] Joel Lehman and Kenneth O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223, 2011.
- [30] Joel Lehman and Kenneth O. Stanley. *Novelty Search and the Problem with Objectives*, pages 37–56. Springer New York, New York, NY, October 2011.
- [31] Thomas Lombardi, Sung-Hyuk Cha, and Charles Tappert. A lightweight image retrieval system for paintings. In *Proceedings of the IS&T/SPIE Electronic Imaging*, volume 5682, pages 236–246, 01 2005.
- [32] Rodolfo A. Lopes, Rodrigo C. Pedrosa Silva, Felipe Campelo, and Frederico G. Guimarães. Dynamic selection of migration flows in island model differential evolution. In *Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation, GECCO '13 Companion*, pages 173–174, New York, NY, USA, July 2013. Association for Computing Machinery.
- [33] Sean Luke. ECJ evolutionary computation library, 1998. Available for free at <http://cs.gmu.edu/~eclab/projects/ecj/>.
- [34] James McClintock and Gary G. Yen. A two-tiered, agent based approach for autonomous, evolutionary texture generation. In *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 3220–3227, 2008.
- [35] Liam McDevitt. A framework for meta-heuristic parameter performance prediction using fitness landscape analysis and machine learning. Master’s thesis, Brock University, 2023.
- [36] Tom McReynolds and David Blythe. Chapter 18 - natural detail. In Tom McReynolds and David Blythe, editors, *Advanced Graphics Programming Using OpenGL*, The Morgan Kaufmann Series in Computer Graphics, pages 467–500. Morgan Kaufmann, San Francisco, 2005.
- [37] Risto Miikkulainen, Hormoz Shahrzad, and Babak Hodjat. Distributed Age-Layered Novelty Search. *ALIFE 2016, the Fifteenth International Conference on the Synthesis and Simulation of Living Systems*, pages 131–138, July 2016.
- [38] Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *CoRR*, abs/1504.04909, 2015.

- [39] Jørgen Nordmoen, Eivind Samuelsen, Kai Olav Ellefsen, and Kyrre Glette. Dynamic mutation in map-elites for robotic repertoire generation. *ALIFE 2018: The 2018 Conference on Artificial Life*, pages 598–605, July 2018.
- [40] Ken Perlin. Improving noise. *ACM Trans. Graph.*, 21(3):681–682, jul 2002.
- [41] Matt Pharr, Wenzel Jakob, and Greg Humphreys. 10 - texture. In Matt Pharr, Wenzel Jakob, and Greg Humphreys, editors, *Physically Based Rendering (Third Edition)*, pages 597–669. Morgan Kaufmann, Boston, third edition edition, 2017.
- [42] Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. *A field guide to genetic programming*. Published via <http://lulu.com> and freely available at <http://www.gp-field-guide.org.uk>, 01 2008. (With contributions by J. R. Koza).
- [43] "Charles Poynton". 29 - component video colour coding for sd.
- [44] Andrew Pozzuoli. Diversifying emergent behaviours with age-layered map-elites. Master's thesis, Brock University, 2023.
- [45] Justin K. Pugh, Lisa B. Soros, and Kenneth O. Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers Robotics AI*, 3:40, 2016.
- [46] Xuejie Qin and Yee-Hong Yang. Estimating parameters for procedural texturing by genetic algorithms. *Graphical Models*, 64(1):19–39, January 2002.
- [47] Rhys. Java structural similarity, April 2018. Accessible online at <https://github.com/rhys-e/structural-similarity>.
- [48] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.
- [49] Aleksander Skakovski and Piotr Jedrzejowicz. An island-based differential evolution algorithm with the multi-size populations. *Expert Systems with Applications*, 126:308–320, July 2019.
- [50] Michael J. Swain and Dana H. Ballard. Color indexing. *Int. J. Comput. Vision*, 7(1):11–32, 11 1991.
- [51] Vassilis Vassiliades, Konstantinos Chatzilygeroudis, and Jean-Baptiste Mouret. Scaling up map-elites using centroidal voronoi tessellations. *IEEE Transactions on Evolutionary Computation*, 22:623–630, August 2017.

- [52] James S. Walker. *A Primer on wavelets and their scientific applications*. Chapman & Hall/CRC, 2 edition, 2008.
- [53] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.
- [54] Li-Yi Wei and Marc Levoy. Texture analysis and synthesis, 09 2022. Accessed online at <https://graphics.stanford.edu/projects/texture/>.
- [55] Darrell Whitley, Soraya Rana, and Robert Heckendorn. The island model genetic algorithm: On separability, population size and convergence. *Journal of Computing and Information Technology*, 7, December 1998.
- [56] Darrell Whitley, Soraya Rana, and Robert B. Heckendorn. Island-model genetic algorithms and linearly separable problems. In David Corne and Jonathan L. Shapiro, editors, *Evolutionary Computing*, pages 109–125, Berlin, Heidelberg, April 1997. Springer Berlin Heidelberg.
- [57] Andrea L. Wiens and Brian Ross. Gentropy: Evolving 2d textures. *Computers and Graphics*, pages 75–88, February 2002.
- [58] Han Zhu and Brian Ross. Procedural texture evolution using multiobjective optimization. *New Generation Computing*, 22:271–293, September 2004.

Appendix A

Additional Experimental Analysis

A.1 Wavelet CHISTQ SSIM Experiment Run Results

Table A.1: IM Subpopulation 1 Image Results

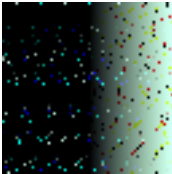
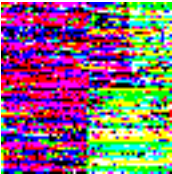
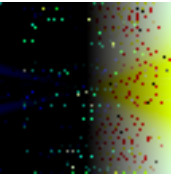
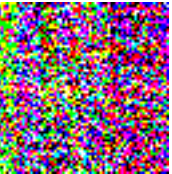
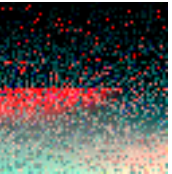
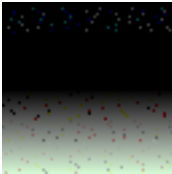
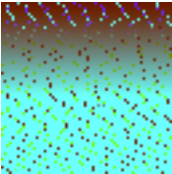

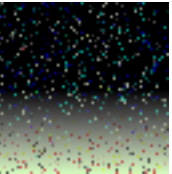
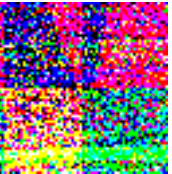
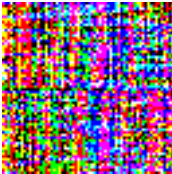
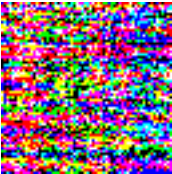

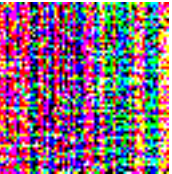
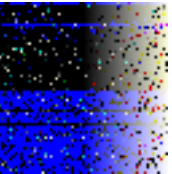
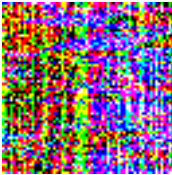
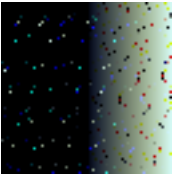
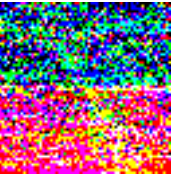
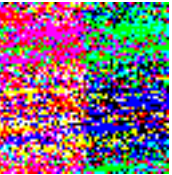
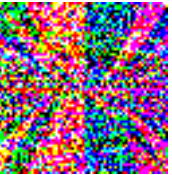
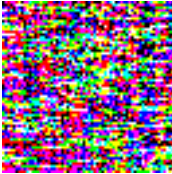

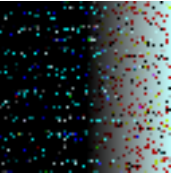
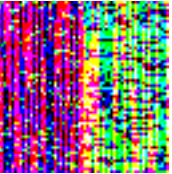
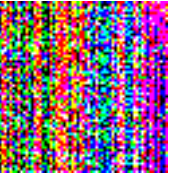
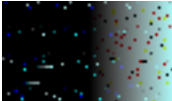
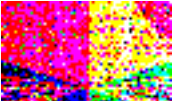

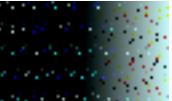

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

Table A.2: IM Subpopulation 2 Image Results




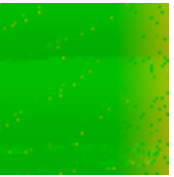
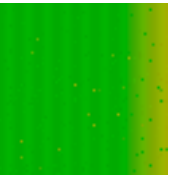



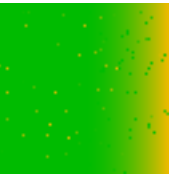





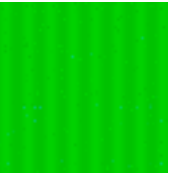


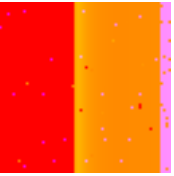

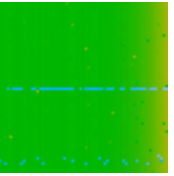







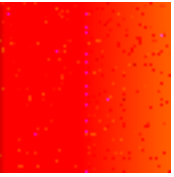
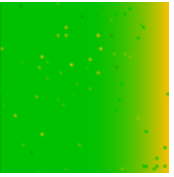

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

Table A.3: IM Subpopulation 3 Image Results


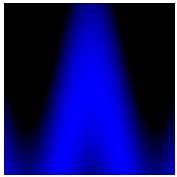
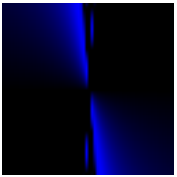
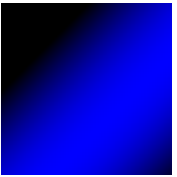
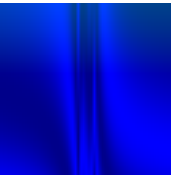

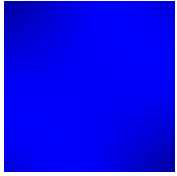

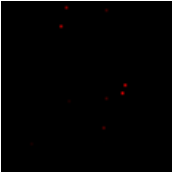
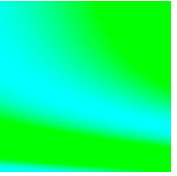

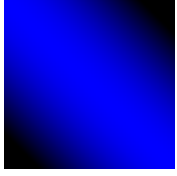
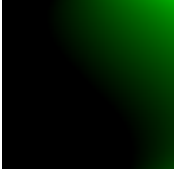
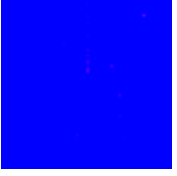
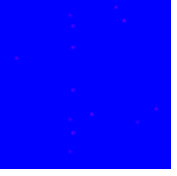
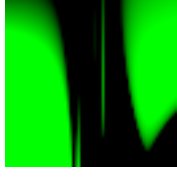
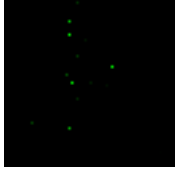
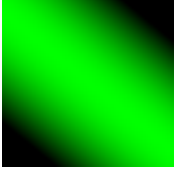
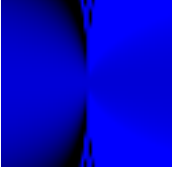
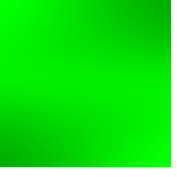
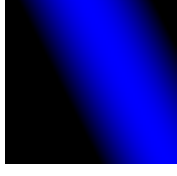
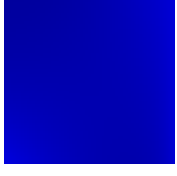

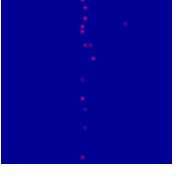
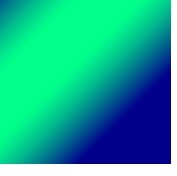

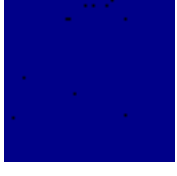
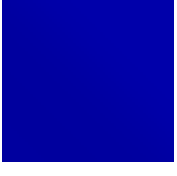


Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

Table A.4: DME Sub-MAP 1 Image Results

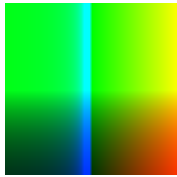
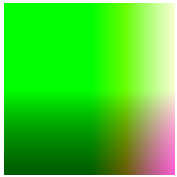
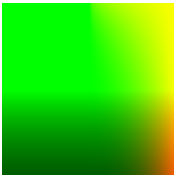

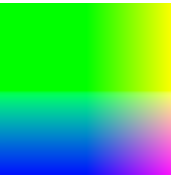

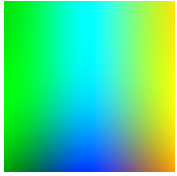
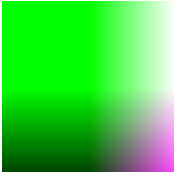
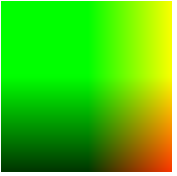
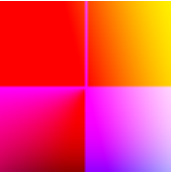
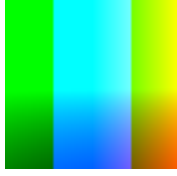
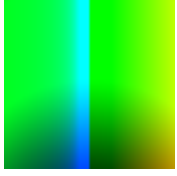
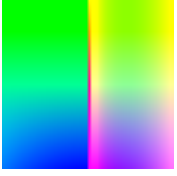
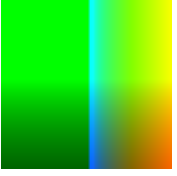
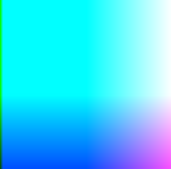

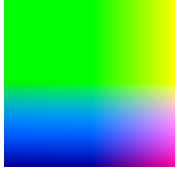
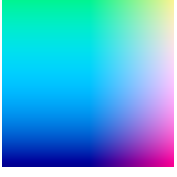



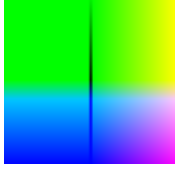
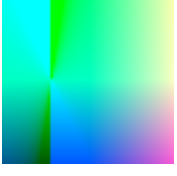

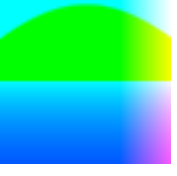
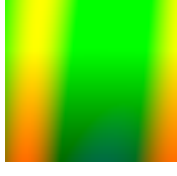
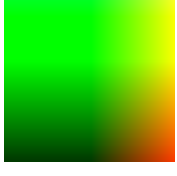
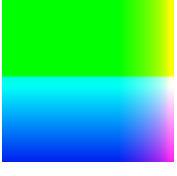
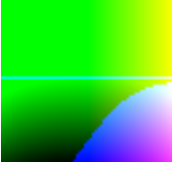
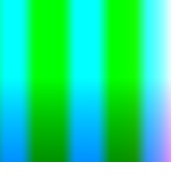
Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

Table A.5: DME Sub-MAP 2 Image Results

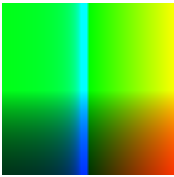
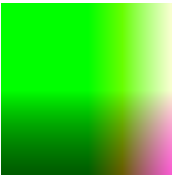
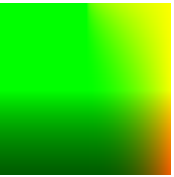

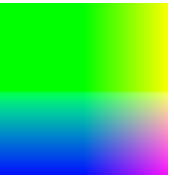
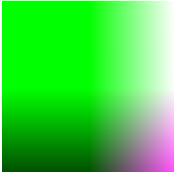
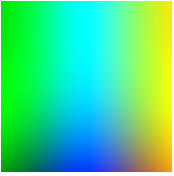
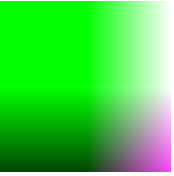
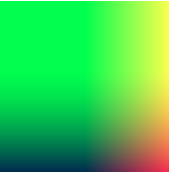
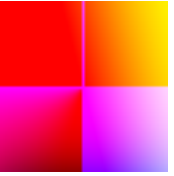

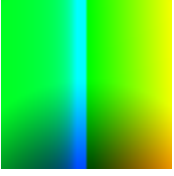
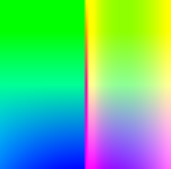
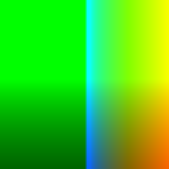
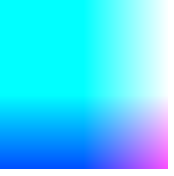

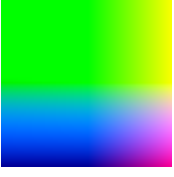

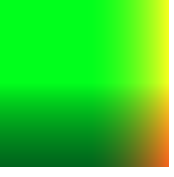


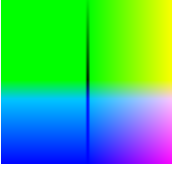
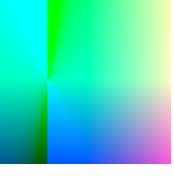

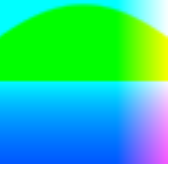
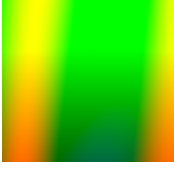
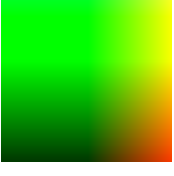

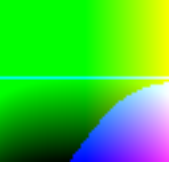
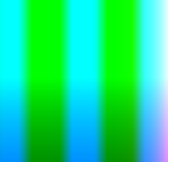
Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
				
				
				
				
				
				

Table A.6: DME Sub-MAP 3 Image Results

Runs 1-6	Runs 7-12	Runs 13-18	Runs 19-24	Runs 25-30
