Discrepancy-Based Evolutionary Diversity Optimization

Aneta Neumann
The University of Adelaide
School of Computer Science
Adelaide, Australia

Wanru Gao
The University of Adelaide
School of Computer Science
Adelaide, Australia

Carola Doerr
Sorbonne Université, CNRS
Laboratoire d'informatique de Paris 6
Paris, France

Frank Neumann
The University of Adelaide
School of Computer Science
Adelaide, Australia

Markus Wagner
The University of Adelaide
School of Computer Science
Adelaide, Australia

ABSTRACT

Diversity plays a crucial role in evolutionary computation. While diversity has been mainly used to prevent the population of an evolutionary algorithm from premature convergence, the use of evolutionary algorithms to obtain a diverse set of solutions has gained increasing attention in recent years. Diversity optimization in terms of features on the underlying problem allows to obtain a better understanding of possible solutions to the problem at hand and can be used for algorithm selection when dealing with combinatorial optimization problems such as the Traveling Salesperson Problem.

We consider discrepancy-based diversity optimization approaches for evolving diverse sets of images as well as instances of the Traveling Salesperson problem where a local search is not able to find near optimal solutions. Our experimental investigations comparing three diversity optimization approaches show that a discrepancy-based diversity optimization approach using a tie-breaking rule based on weighted differences to surrounding feature points provides the best results in terms of the star discrepancy measure.

CCS CONCEPTS

• **Theory of computation → Evolutionary algorithms**;

KEYWORDS

Diversity, evolutionary algorithms, features

ACM Reference Format:
Aneta Neumann, Wanru Gao, Carola Doerr, Frank Neumann, and Markus Wagner. 2018. Discrepancy-Based Evolutionary Diversity Optimization. In *GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205532

1 INTRODUCTION

Diversity plays a crucial role in evolutionary computation. Traditionally, diversity is used to avoid premature convergence and it is generally assumed that crossover-based evolutionary algorithms need a diverse population in order to produce good results.

During the last 10 years, using evolutionary algorithms to produce a diverse set of solutions has gained increasing attention. Ulrich and Thiele [30] introduced evolutionary computation approaches that are able to produce diverse sets of solutions by evolving a population according to quality criteria and diversity measures.

Recently, this approach has been adapted to evolve diverse sets of Traveling Salesperson Problem (TSP) instances [11] as well as diverse sets of images [1]. In the case of the TSP, instances have been evolved that are hard to be solved by a given solver. In this case, diversity is measured according to different features that characterize the problem instances. In the case of images, the population of an evolutionary algorithm has been used to evolve images that are close to a given one (in terms of an error measure) and that are diverse with respect to different artistic features. Furthermore, an evolutionary image composition approach based on a feature-based covariance error function has been introduced in [22]. Both diversity optimization approaches build on a simple diversity measure that measures diversity according to a given feature. In order to extend this approach to more than one feature, a diversity measure weightening the different features has been used.

In this paper, we introduce a diversity optimization approach using the discrepancy measure. This approach allows to evolve diverse sets without having any assumption on the preferred weightening of the different diversity criteria.

Discrepancy theory studies the *irregularity of distributions* in the following sense. Given a metric space S and some n points $s_1, \ldots, s_n \in S$, the discrepancy of the set $X := \{s_1, \ldots, s_n\}$ is measured as the largest deviation from a perfectly evenly distributed point set. When, as in our case, $S = [0,1]^d$ is the d-dimensional unit cube, we could measure the discrepancy with respect to all axis-parallel boxes $[a,b] := [a_1,b_1] \times \ldots \times [a_d,b_d]$. In an ideal situation, we would like the number of points of X that are inside such a box $[a,b]$ to be proportional to its volume. In other words, we would like the difference $\mathrm{Vol}([a,b]) - |X \cap [a,b]|/n$ to be as small as possible, simultaneously for all possible boxes $[a,b]$. The discrepancy is set to be the largest deviation; i.e.,

$$D(X, \mathcal{B}) := \sup\{\mathrm{Vol}([a,b]) - |X \cap [a,b]|/n \mid a \leq b \in [0,1]^d\},$$

where we abbreviate $a \leq b$ if and only if for every component $i \in d$ the inequality $a_i \leq b_i$ holds. The smaller the discrepancy of a point set, the more regular is its distribution with respect to all axis-parallel boxes.

Discrepancy theory plays an important role in numerical integration, where (under certain circumstances), low discrepancy point sets are known to provide very good estimates for the integral of an unknown or difficult-to-analyze function. Classical Monte Carlo integration is therefore often replaced by a so-called Quasi-Monte Carlo integration, which uses low discrepancy point sets instead of purely random ones, cf. [18] for an illustrated introduction to discrepancy theory. In the context of evolutionary computation, low discrepancy points sets such as Sobol and Halton sequences have been used in the sampling routines of evolution strategies [3, 25], CMA-ES variants [27–29], and other genetic algorithms [15, 16], and are reported to bring efficiency gains over pure random sampling. On the other hand, evolutionary algorithms have been used to compute point sets of low discrepancy values [5, 10], an optimization problem not admissible by traditional analytical approaches. Finally, randomized search heuristics also play a crucial role for the computation of discrepancy values of point sets in high dimensions [13].

The arguably most intensively studied discrepancy notion is the so-called *star discrepancy*, which measures the regularity with respect to all axis-parallel boxes $[0, b]$, $b \in [0, 1]^d$ that are anchored in the origin. This is also the measure for which Sobol and Halton sequences have been designed for. Here in this work, we use this star discrepancy measure to evaluate how evenly the points are distributed.

At first sight, one might conjecture that a regular $\sqrt{n} \times \sqrt{n}$ grid has a good and regular distribution. Its star discrepancy, however, is rather large: we easily convince ourselves there are boxes of volume $1/\sqrt{n}$ which do not contain any point, so that the star discrepancy is of at least this order. Random point sets also achieve a discrepancy value of order $1/\sqrt{n}$ only. In contrast, the low-discrepancy sequences mentioned above achieve a discrepancy value of order $\log^{d-1}/n$, and are thus much more evenly distributed with in terms of discrepancy.

Apart from numerical integration and the mentioned applications in evolutionary computation, low discrepancy sequences also play an important role in statistics, computer graphics, and stochastic programming.

We investigate the use of the star discrepancy measure in evolutionary diversity optimization for two settings previously studied in the literature, namely diversity optimization for images [1] and TSP instances [11]. In terms of images, we also introduce a new and more effective mutation operator based on random walks than the one introduced in [1]. This self-adaptive random walk operator allows to reduce the number of iterations needed to construct good diverse set of solutions from 1 – 4 million [1] to 2000 and therefore reduces the number of required generations by three orders of magnitude.

Our experiments are carried out for diversity optimization tasks using two and three features. We show that the previously used approach for images [1] and TSP instances [11] computing a weighted diversity contribution in terms of the considered features constructs solution sets with a very high discrepancy compared to our approach using the discrepancy measure. Furthermore, we show that the weighted diversity contribution approach can be used in an effective way for doing tie-breaking between sets of solutions having the same discrepancy value.

The paper is structured as follows. In Section 2, we introduce our discrepancy-based diversity optimization approach. In Section 3, we introduce the new mutation operator for diversity optimization of images and evaluate the discrepancy optimization approach for images. We consider our approach for evolving sets of TSP instances of low discrepancy with respect to the given features in Section 4. Finally, we finish with some concluding remarks.

Algorithm 1: $(\mu + \lambda)$-EA_D

```
1 Initialize the population $P$ with $\mu$ instances of quality at least $\alpha$.
2 Let $C \subseteq P$ where $|C| = \lambda$.
3 For each $I \in C$, produce an offspring $I'$ of $I$ by mutation. If
    $q(I') \geqslant \alpha$, add $I'$ to $P$.
4 While $|P| > \mu$, remove an individual $I = \arg\min_{J \in P} D^*(P \setminus J)$.
5 Repeat step 2 to 4 until termination criterion is reached.
```

2 DISCREPANCY-BASED DIVERSITY OPTIMIZATION

We consider evolutionary diversity optimization. Given a search space S, our aim is to construct a diverse set of solutions $P = \{X_1, \ldots, X_\mu\}$ where each solution $X_i \in S$ fulfills a given quality criteria, i.e., we have $q(X_i) \geq \alpha$ for a given quality threshold α.

Properties of our potential solutions X_i are characterized by features $f_1, \ldots, f_d$ which are problem specific. Let $I \in S$ be an individual in a population P. We associate with I its feature vector $f(I) = (f_1(I), \ldots, f_d(I))$.

Traditionally, the goal of constructing a set of points with a low discrepancy is defined in $[0, 1]^d$. Therefore, the feature values are scaled before the calculation of discrepancy. Let $f_i^{\max}$ and $f_i^{\min}$ be the maximum and minimum value of feature f_i. We evaluate our set of points in terms of discrepancy using the scaled feature values

$$f_i'(I) = (f_i(I) - f_i^{\min})/(f_i^{\max} - f_i^{\min}).$$

We have $f'(I) \in [0, 1]^d$ for all scaled feature vectors $f'(I)$ if $f_i^{\min} \leq f_i(I) \leq f_i^{\max}$, $1 \leq i \leq d$. $f_{\max}$ and $f_{\min}$ are set based on initial experiments. Feature values outside that range will be scaled to 0 and 1, respectively, to allow the algorithm to work with non anticipated features values.

Let $f'(P) = \cup_{I \in P} f'(I)$ be the set of scaled features vectors in P. We denote by $D^*(P)$ the discrepancy of $f'(P)$ in $[0, 1]^d$. Throughout this paper, we use the star-discrepancy. Given $P = \{I_1, \ldots, I_k\}$ with feature vectors $f'(I_1), \ldots, f'(I_k)$, we define

$$D^*(P) := \sup_{J \in Y} D(J, P),$$

where

$$D(J, P) = \frac{|I \in P \mid f'(I) \in J|}{k} - \mathrm{Vol}(J),$$

Vol(J) denotes the volume of the box J, and Y is the class of all axis-parallel boxes of the form $J = \prod_{i=1}^{d}[0, u_i)$ with $0 \le u_i \le 1$ for $1 \le i \le d$.

A key difficulty to overcome in the optimization for low star discrepancy values is the computational hardness of its evaluation [12]. The best known algorithm for the star discrepancy computation has a running time of order $n^{1+d/2}$ [7], which is exponential in the dimension d. As we are interested in dimension $d = 2, 3$, we can use this algorithm, and make use of the implementation that is available on [31]. The reader interested in a discussion of computational aspects of geometric discrepancies is referred to [9].

We use the $(\mu + \lambda)$-EA_D given in Algorithm 1 to compute a diverse population where each individual meets a given quality criteria q according to a given threshold α, i.e., we have $q(I) \ge \alpha$ for all individuals in the population P. The population P is a multi-set, i.e., it may contain an instance more than once. The algorithm is initialized with a population where each individual meets the given criteria. In each iteration λ offspring are produced. Offspring that do not meet the quality criteria are directly discarded. Offspring that meet the criteria are added to the population and survival selection is performed afterwards to obtain a population of size μ. To do this, individuals are removed iteratively. Having a population of size $k > \mu$, in each iteration an individual I is removed that leads to a population $P \setminus I$ of size $k-1$ having the smallest discrepancy among all populations that can be constructed by removing exactly one individual from P.

The discrepancy minimization algorithm is compared to the evolutionary diversity optimization approach in [11], which aims at maximizing the feature-based population diversity using a weighted contribution measure for each individual. The weighted diversity contribution of an individual I with feature vector $f(I)$ is defined as $c(I, P) = \sum_{i=1}^{k}(w_i \cdot d_{f_i}(I, P))$, where $d_{f_i}(I, P)$ represents the normalised contribution of individual I to the population diversity over feature f_i and w_i represents the weight for feature f_i.

The resulting algorithm $(\mu+\lambda)$-EA$_C$ differs from $(\mu+\lambda)$-EA$_D$ only in step 4, and removes in each of these steps an individual I with the smallest weighting contribution $c(I, P)$ to the population diversity. Furthermore, we consider the algorithm $(\mu + \lambda)$-EA$_T$ that uses both the discrepancy measure and the weighted contribution measure. It is the same as $(\mu + \lambda)$-EA$_D$ but uses the weighted contribution measure as tie-breaking in step 4 of the algorithm; i.e. if there is more than one individual whose removal leads to the minimum discrepancy value, then the one among these with the smallest contribution to the weighted diversity contribution is removed.

In the following, we evaluate our discrepancy-based diversity optimization approaches for evolving diverse sets of images and TSP instances. We also introduce a new mutation operator for images based on random walks which significantly speeds up the diversity optimization process when constructing a diverse set of images.

3 IMAGES

We consider the task of evolving a diverse set of images as previously investigated in [1]. Given an image S, the task is to compute a diverse set of images $P = \{I_1, \ldots, I_\mu\}$ that meets a given quality criterion $q(I)$ for each $I \in P$. For our experimental investigations, an image I meets the quality criterion if the *mean-squared error* in terms of the RGB-value of the pixels of the image I with respect to the input image S (shown in Figure 1) is less than 500.

Figure 1: Image S

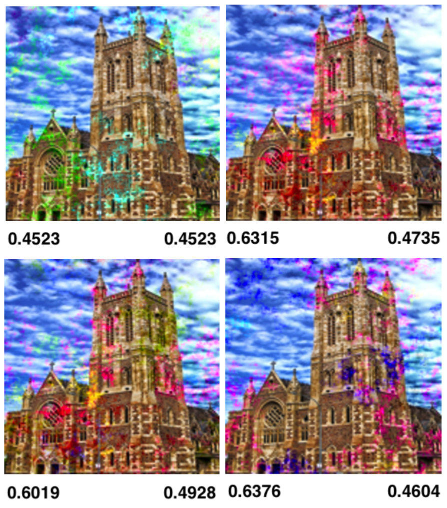

Figure 2: Selected images from the population after discrepancy minimization for the Hue and Saturation features.

Many features have been used to measure the characteristics of images. We focus on a selected set of features used in [1, 6],*standard-deviation-hue (SD-hue)*, *mean-saturation*, *reflectional symmetry* [6], *mean-hue*, *Global Contrast Factor (GCF)* [17] and *smoothness* [23]. We carry out our discrepancy-based diversity optimization approach for these different features and use the evolutionary algorithm to evolve diverse populations of images for each feature combination.

We focus our experiments on the characterization of how the chosen features may influence the generated images. In reference to previous work [21] we choose three pairs of features: (*SD-hue*, *mean-saturation*), (*symmetry*, *mean-hue*), and (*GCF*, *smoothness*). In addition, we choose three feature sets consisting of three features each as follows: (*SD-hue*, *mean-saturation*, *symmetry*) and (*SD-hue*, *mean-hue*, *symmetry*), and (*GCF*, *mean-hue*, *mean-saturation*).

We are working with the scaled feature values when computing the discrepancy of a given set of points. It should be pointed out that not all feature vector combinations within the given feature intervals are usually possible. To illustrate this we consider the features SD-hue and Saturation and run the EA (using the mutation operator described in Section 3.1) for 1000 iterations. Figure 3 shows all feature vectors produced during 10 runs of the $(20 + 1)$-EA_T

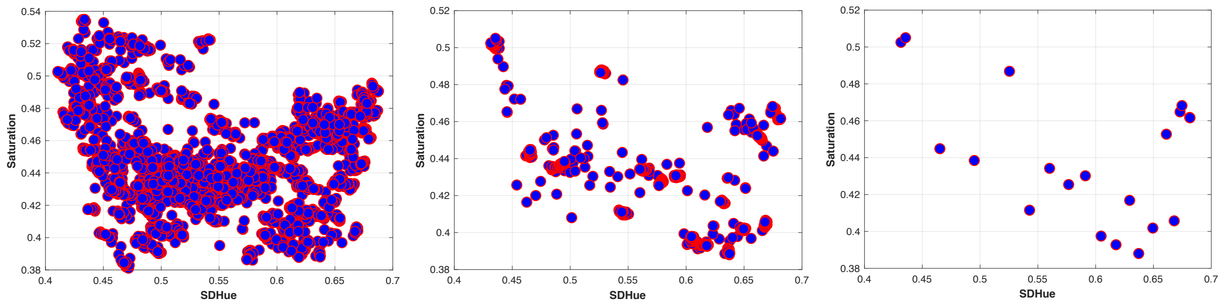

Figure 3: All feature vectors generated in 10 runs of $(\mu + \lambda)$-EA_T with 1000 iterations each (left), one run with 1000 iterations (middle), the final population after 1000 iteration with discrepancy 0.22637 (right).

Algorithm 2: OffsRandomWalkMutation $(X, t_{\max})$

```
1  Let $X$ is a image with pixels $X_{ij} \in X$.
2  $Y \leftarrow X$.
3  Choose starting pixel $Y_{ij} \in Y$ uniformly at random.
4  Choose offset $o \in [-r, r]^3$ uniformly at random.
5  $t \leftarrow 1$.
6  while $t \leq t_{\max}$ do
7      $Y_{ij} = Y_{ij} + o$.
8      Choose $Y_{kl} \in N(Y_{ij})$ uniformly at random.
9      $i \leftarrow k, j \leftarrow l$.
10     $t = t + 1$.
11 Return $Y$.
```

(left), all feature vectors produced during one run (middle), and the feature vectors of the final population (right). It can be observed that the area where both feature values are high does not contain any points (similarly if both feature values are very low). The seems to indicate that the problem is constrained to a subspace of the unit square. If this is true, then this has a direct consequence on the best possible discrepancy value that can be obtained, as discrepancy is a measure in $[0, 1]^d$.

3.1 Self-Adjusting Offset Random Walk Mutation

The algorithm uses a variant of the random walk mutation introduced in [22] for evolutionary image composition. This speeds up the process of diversity optimization by three orders of magnitude compared to [1] where for a mutation operator changing in each step a single pixel 1–4 million iterations where required to construct a diverse set of images. Our new mutation operator enables us to construct diverse sets of images for all three algorithms (including the $(\mu + \lambda)$-EA$_C$ investigated in [1]) within just 2000 generations.

The random walk in this paper differs from the one for image composition given in [22] by changing the *RGB* values by an offset vector $o \in [-r, r]^3$ chosen in each mutation step uniformly at random. The mutation operator is shown in the Algorithm 2.

The random walk causes movement from the current pixel X_{ij} to the next pixel by moving either right, left, down or up. We define the neighborhood $N(X_{ij})$ of pixel X_{ij} as

$$N(X_{ij}) = \left\{X_{(i-1)j}, X_{(i+1)j}, X_{i(j-1)}, X_{i(j+1)}\right\}.$$

The random walk chooses an element of $N(X_{ij})$ uniformly at random in every step. Furthermore, the random walk is wrapped around the boundaries of the image. We produce an offspring Y from X by setting each visited pixel X_{ij} to the value of $X_{ij} + o$. Given a current image X, our $(\mu + \lambda) - EA_D$ algorithm uses the random walk mutation to alter all visited pixels. Note that pixels may be visited more than once and the offset may be applied several times in this case. The random walk paints all the visited pixels by adding the chosen offset vector o. Each random walk mutation is run for $t_{\max}$ steps, where $t_{\max}$ is chosen in an adaptive way.

3.1.1 *Self-Adjustment.*

We decrease the length of random walks through decreasing $t_{\max}$ when the discrepancy value does not decrease as a result of an unsuccessful mutation. We increase $t_{\max}$ if the discrepancy decreases as a result of a successful mutation. This builds on the assumption that mutations doing less change to the image are needed to obtain an improvement if it is hard to make progress with the current choice of $t_{\max}$. On the other hand, a better progress may be achievable if the current setting of $t_{\max}$ is already able to decrease the discrepancy. Our adaptive approach makes use of the parameter adjusting scheme recently used in [8]. This method, originally proposed in [14], applies the classical 1/5-success rule from evolution strategies to a discrete setting.

Our approach increases $t_{\max}$ for a successful outcome or decreases $t_{\max}$ in the case that the new offspring is not accepted. In our algorithm, $t_{\max}$ can take on values in $t_{\mathrm{LB}} \leq t_{\max} \leq t_{\mathrm{UB}}$, where t_{LB} is a lower bound on $t_{\max}$ and t_{UB} is an upper bound on $t_{\max}$.

For a successful mutation, we set $t_{\max} := \min\{F \cdot t_{\max}, t_{\mathrm{UB}}\}$ and for an unsuccessful mutation, we set $t_{\max} := \max\left\{F^{-1/k} \cdot t_{\max}, t_{\mathrm{LB}}\right\}$, where $F > 1$ is a real value and $k \geq 1$ an integer which determines the adaptation speed.

For our experimental investigations, we set $t_{\mathrm{LB}} = 1000$, $t_{\mathrm{UB}} = 20000$, $F = 2$, $k = 8$, and $t_{\max} = 1000$ at initialization based on preliminary experimental investigations.

3.2 Experimental settings

All algorithms were implemented in *Matlab* (*R*2017*b*). We ran all of our experiments on single nodes of a Lenovo NeXtScale M5 Cluster

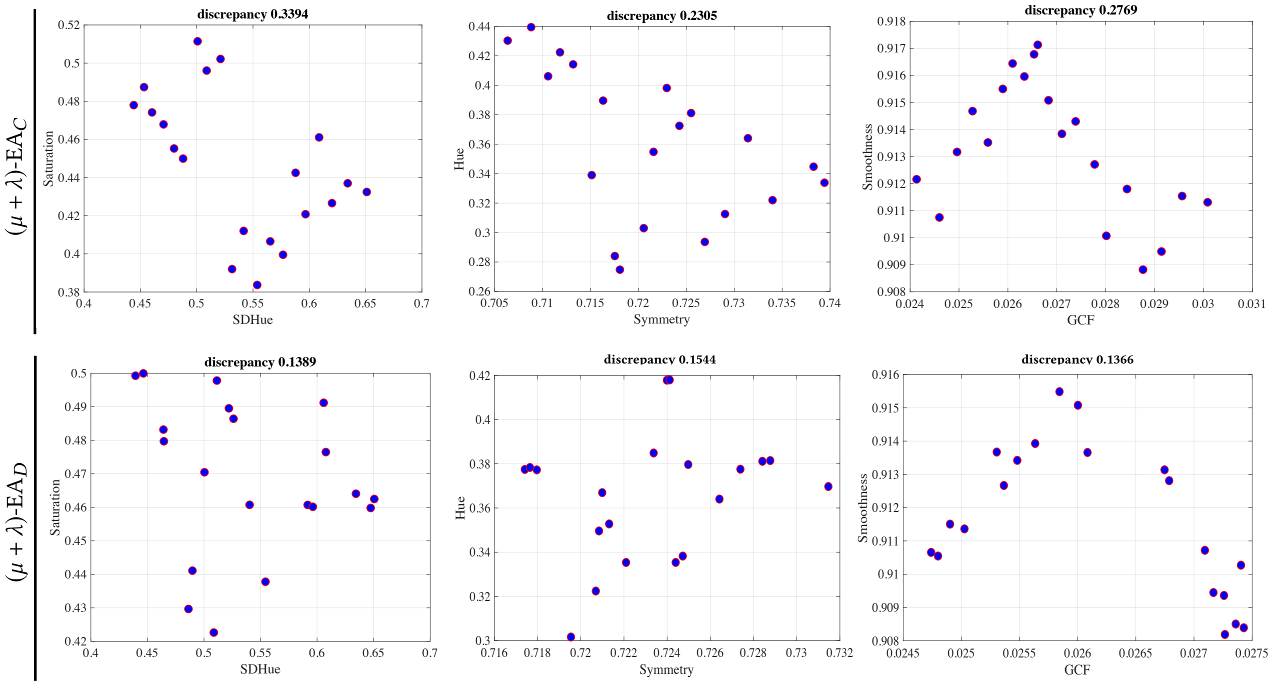

Figure 4: Feature vectors for final population of $(\mu + \lambda)$-EA$_C$ (top) and $(\mu + \lambda)$-EA$_D$ (bottom) for images based on two features from left to right: (SDHue, Saturation), (Symmetry, Hue), (GCF, Smoothness).

	$(\mu+\lambda)$-$EA_C(1)$				$(\mu+\lambda)$-EA_D (2)				$(\mu+\lambda)$-$EA_T(3)$			
	min	mean	std	stat	min	mean	std	stat	min	mean	std	stat
(f1, f2)	0.2014	0.3234	0.0595	$2^{(-)}$,$3^{(-)}$	0.1272	0.2038	0.1157	$1^{(+)}$	0.1119	0.1530	0.0269	$1^{(+)}$
(f3, f4)	0.1964	0.2945	0.0497	$2^{(-)}$,$3^{(-)}$	0.1574	0.2280	0.0592	$1^{(+)}$,$3^{(-)}$	0.1051	0.1417	0.0179	$1^{(+)}$,$2^{(+)}$
(f5, f6)	0.1997	0.2769	0.0344	$2^{(-)}$,$3^{(-)}$	0.1363	0.2025	0.0538	$1^{(+)}$	0.1457	0.1800	0.0234	$1^{(+)}$
(f1, f2, f3)	0.3389	0.4327	0.0613	$2^{(-)}$,$3^{(-)}$	0.1513	0.3335	0.1062	$1^{(+)}$	0.2253	0.2814	0.0422	$1^{(+)}$
(f1, f4, f3)	0.2754	0.3395	0.0483	$2^{(-)}$,$3^{(-)}$	0.2100	0.3118	0.1309	$1^{(+)}$	0.2224	0.2600	0.0123	$1^{(+)}$
(f5, f4, f2)	0.4775	0.6488	0.0841	$2^{(-)}$,$3^{(-)}$	0.2021	0.3007	0.1467	$1^{(+)}$	0.1983	0.2229	0.0125	$1^{(+)}$

Table 1: Statistics of discrepancy values for images. f1, f2, f3, f4, f5, f6 denote features SD-hue, Saturation, Symmetry, Hue, GCF and Smoothness, respectively.

with two Intel Xeon E5−2600 v4 series 16 core processors, each with 64GB of RAM.

Firstly, we consider the discrepancy-based diversity optimization for two features. We select features in order to combine different aesthetic and general features based on our initial experimental investigations and previous investigations in [21]. Furthermore, we set $f^{\min}$ and $f^{\max}$ as follows. The $f^{\min}$ values used for *SD-hue*, *Hue*, *Saturation*, *Smoothness*, *GCF*, *Symmetry* are 0.42, 0.25, 0.42, 0.42, 0.906, 0.0245, and 0.715, respectively. The corresponding $f^{\max}$ values are 0.7, 0.4, 0.5, 0.5, 0.918, 0.0275, and 0.74, respectively.

After considering the combination of two features, we investigate sets of three features. Here, we select different features combining aesthetic and general features together used in the previous experiment. In order to obtain a clear comparison between experiments, we apply the same range of feature values as before.

Furthermore, we run the $(\mu + \lambda)$-EA_C diversity algorithm from [21] using the self-adjusting random walk mutation operator in order to compare the two approaches for diversity optimization. We use the same settings for the $(\mu+\lambda)$-EA_C as for our discrepancy-based diversity algorithm, the $(\mu+\lambda)$-EA_D. Finally, we consider the $(\mu + \lambda)$-EA_T, which uses discrepancy-based diversity optimization plus tie-breaking according to weighted feature contributions when more than one individual exists whose removal would result in the same minimal discrepancy value.

We run each algorithm for 2000 generations with a population size of $\mu = 20$ and $\lambda = 1$. In order to evaluate our results using statistical tests, each algorithm is run 30 times with the same setting applied to each considered pair and triple of features.

3.3 Experimental Results

We perform a series of experiments to evaluate the performance of our discrepancy-based diversity evolutionary algorithm. Our experiments establish that global constraints like *mean-squared* *error* can be used to produce more diverse images than equivalent constraints which are limited to the range of the color or luminosity channel for each pixel.

The images displayed in Figure 2 match the color features. In particular, images are produced using the *SDHue* and *Saturation* feature. The values for *SDHue* and *Saturation* are displayed above each image for features, respectively. The color spectrum is red at each end with individuals of the population spread along it. Images which have a low score for this feature will be monochromatic and will appear in the middle of the spectrum. Images with a high score will be red as it is a sample from both of the extremes. Low-scoring images in the *Saturation* feature are monochromatic whilst high-scoring individuals are almost entirely saturated.

Figure 4 shows feature plots of the final populations of the $(\mu+\lambda)$-EA_C (top) and the $(\mu+\lambda)$-EA_D (bottom) for 3 pairs of feature combinations. It can be observed that the discrepancy value for the $(\mu+\lambda)$-EA_D and the combination $(SD - hue, Saturation)$ is 0.1389. This is significantly smaller than the one for the $(\mu+\lambda)$-EA_C at 0.3394. The middle row shows the combination of *Hue* and *Symmetry*. The discrepancy value of *Symmetry* and *Hue* for $(\mu+\lambda)$-EA_D is 0.1544 whereas it is 0.2305 for $(\mu+\lambda)$-EA_C. In Figure 4 the right column shows the final populations of the diversity optimization when considering *GCF* and *Smoothness*. The discrepancy value for *GCF* and *Smoothness* is 0.1366 for $(\mu+\lambda)$-EA_D and 0.2769 for $(\mu+\lambda)$-EA_C. This is an indication of the difficulty of evolving images which are smooth as well as scoring high in *GCF* in our current setup. However, this conflict is expected as the *GCF* highly scores in the case of strong contrast between adjacent pixels. The *Smoothness* scores have a high value for low contrast between neighboring pixels.

We now consider the results of the $(\mu+\lambda)$-EA_C from [21] using the self-adjusting random walk mutation operator in greater detail. Looking at Figure 4 (top) which shows the population of instances for (*SDHue*, *Saturation*), (*Symmetry*, *Hue*), and (*GCF*, *Smoothness*), respectively, we observe that the distribution of the points for the features vectors for final population follows a linear pattern. This is due to the chosen weights which favor lines of feature vectors orthogonal to the used weight vector $(1, 1)$.

We use the Kruskal-Wallis test with 95% confidence in order to measure the statistical validity of our results. We apply the Bonferroni post-hoc statistical procedure that is used for multiple comparison of a control algorithm to two or more algorithms. For more detailed description on the statistical tests we refer the reader to [4]. In Table 1 we provide statistics on the discrepancy values for the final populations of $(\mu+\lambda)$-EA_C, $(\mu+\lambda)$-EA_D and $(\mu+\lambda)$-EA_T, respectively. For each algorithm and feature combination the minimum, mean, and standard deviation of the discrepancy value of the final population of 30 runs is shown. $X^{(+)}$ is equivalent to the statement that algorithm in the column outperformed algorithm X, and $X^{(-)}$ is equivalent to the statement that X outperformed the algorithm given in the column. In the case if the algorithm X not appears this means that no significant difference was determined between algorithms. $(\mu+\lambda)$-EA_D clearly outperforms the $(\mu+\lambda)$-EA_C for all feature combinations. Furthermore, $(\mu+\lambda)$-EA_T which uses tie-breaking according to weighted feature contribution leads to a further improvement of $(\mu+\lambda)$-EA_D for almost all feature combinations in terms of the mean and minimal discrepancy value achieved within 30 runs.

4 TRAVELLING SALESMEN PROBLEM

Another problem we considered as application of Algorithm 1 is the Traveling Salesman Problem (TSP), which is a NP-hard combinatorial optimization problem with many real world applications. We consider the classical Euclidean TSP, which takes a set of cities in the Euclidean plane and where the goal is to find a Hamiltonian cycle with the minimal sum of edge distances.

In this research we focus on TSP instances in the space of $[0, 1]^2$ with 50 cities, which is a reasonable size of problem for feature analysis of TSP. The instances are qualified with respect to the approximation ratio, which is calculated by $\alpha_A(I) = A(I)/OPT(I)$ where $A(I)$ is value of the solution found by algorithm A for the given instance I, and $OPT(I)$ is value of an optimal solution for instance I that is calculated using the exact TSP solver Concorde [2]. Within this study, $A(I)$ is the tour length obtained by three independent repeated runs of the 2-OPT algorithm for a given TSP instance I.

Following the same setting as in [11], the approximation ratio threshold for hard TSP instance of size 50 is set to 1.18, which means only instances with approximation ratio equal or greater than 1.18 are accepted into the population.

As we shall discuss in the next section, a particular interest is the relationship between problem hardness and the feature values [19, 20, 26].

4.1 Experiments settings

There have been many features designed for the TSP with the aim of describing the hardness and characteristics of a certain TSP instance. In this paper, we focus on a selected set of feature values described in [19]. In line with [11] we focus on the following combination of features:

- *angle_mean*: the mean value of the angles made by each point with its two nearest neighbor points
- *mst_depth_mean*: the mean depth of the minimum spanning tree in the TSP
- *centroid_mean_distance_to_centroid*: the mean value of the distances from the points to the centroid
- *mst_dists_mean*: the mean distance of the minimum spanning tree

Note that, while multiple MSTs can exist in principle, we only consider the one returned by the R function spantree, implementing Prim's method.

As mentioned in Section 2, the feature values are normalized before discrepancy calculation. The maximum and minimum values $f^{\max}$ and $f^{\min}$ for each feature are determined based on the results gathered from initial runs of feature-based diversity maximization. The $f^{\max}$ used for the feature angle_mean, centroid_mean_dist_centroid, nnds_mean and mst_dists_mean are 2.8, 0.6, 0.7 and 0.15, respectively. The corresponding $f^{\min}$ values are 0.8, 0.24, 0.1 and 0.06.

Different combinations of features are tested in this research. The algorithms are designed to work with multiple features. As experiment, we choose three two-feature combinations and three

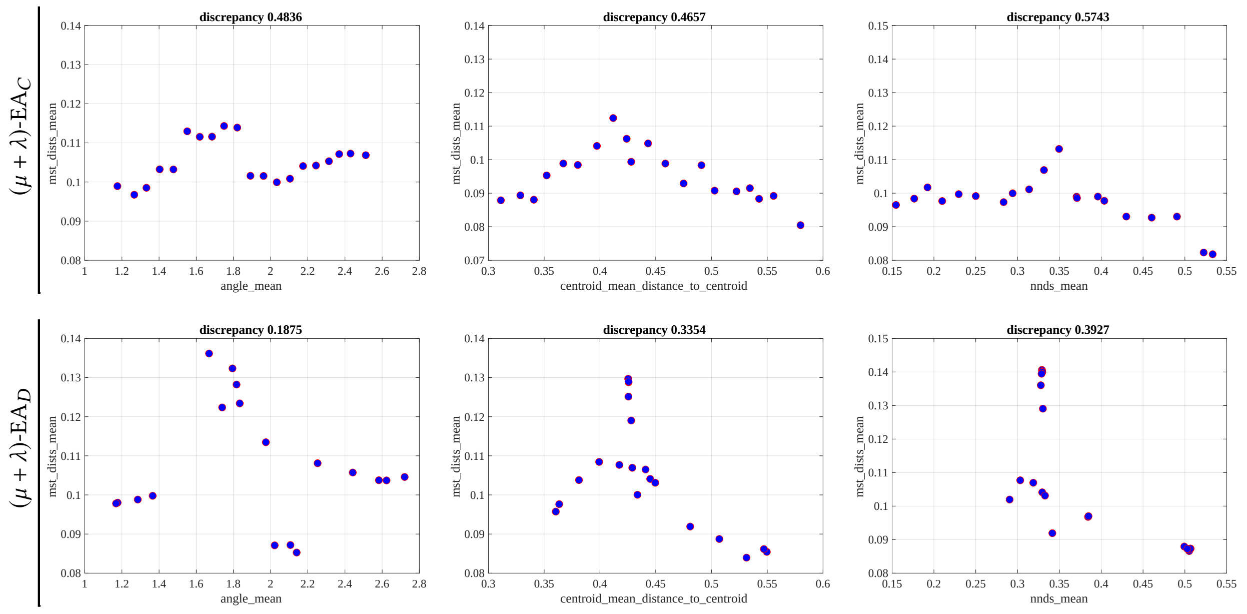

Figure 5: Feature vectors for final population of $(\mu + \lambda)$-EA$_C$ (top) and $(\mu + \lambda)$-EA$_D$ (bottom) for TSP based on two feature from left to right: (angle_mean, mst_dists_mean), (centroid_mean_distance_to_centroid, mst_dists_mean), (nnds_mean, mst_dists_mean)

.

three-feature combinations which are good combinations for classifying problem hardness suggested in [11].

All three algorithms are implemented in R and run in R environment [24]. We use the functions in tspmeta package to compute the feature values [19]. All of the experiments are executed on a machine with 48-core AMD 2.80 GHz CPU and 128 GByte RAM.

Each algorithm is run for 20 000 generations and the final discrepancy is reported. In order to obtain statistics, each feature combination is tested with each algorithm for 30 times. These 30 runs are independent to each other.

4.2 Experimental results and analysis

Figure 5 shows the final population of TSP instances from the run that gets the minimum discrepancy value out of the 30 runs after applying Algorithm $(\mu + \lambda)$-EA_D and $(\mu + \lambda)$-EA_C in the feature space. The average initial discrepancy values for each feature combination in Table 2 are 0.5786, 0.6090, 0.7227, 0.7997, 0.8142 and 0.7699, respectively.

The bottom row of Figure 5 shows the feature vectors for the final population of the $(\mu+\lambda)$-EA_D. Compared to their counterparts in the top row, the discrepancy minimization approach generates a more diverse set for the feature combination of angle_mean and mst_dist_mean. For the feature combination shown in the middle and on the right, it is not so obvious which algorithm generates a more diverse population than the other in the feature space. Each approach obtains a population that explores more over one feature value. For example, the $(\mu + \lambda)$-EA_D generates a population more diverse with respect to the feature of mst_dists_mean, while the $(\mu + \lambda)$-EA$_C$ focuses more on exploring the feature space of centroid_mean_distance_to_centroid. Looking at the discrepancy values, it can be observed that the final population obtained by the $(\mu + \lambda)$-EA_D has a significantly smaller discrepancy than the one obtained by the $(\mu + \lambda)$-EA$_C$ for all 3 pairs of features.

Table 2 shows the statistics about the discrepancy values of the final populations after running each of the three algorithms on three 2-feature combinations and three 3-feature combinations.

The first two large columns contains the statistical results from $(\mu + \lambda)$-EA_C and $(\mu + \lambda)$-EA_D. The $(\mu + \lambda)$-EA_D significantly outperforms the $(\mu + \lambda)$-EA_C in all feature combinations. The average discrepancy value is reduced by more than 30% in all six cases.

During the discrepancy minimization process, there exist many individuals which have the same least contribution to the discrepancy value in each iteration. Breaking ties according to the weighted feature contribution can help to improve the discrepancy of the population. The $(\mu + \lambda)$-EA_T provides breaking ties with respect to the contribution to the weighting population diversity. The third column in Table 2 shows the respective statistics for the $(\mu + \lambda)$-EA_T. For the statistics, it shows that the $(\mu + \lambda)$-EA_T is able to improve the discrepancy values of the final population. In five out of six examined feature combinations, the $(\mu + \lambda)$-EA_T achieves smaller discrepancy values than the $(\mu + \lambda)$-EA_D. For the first two two-feature combinations, $(\mu + \lambda)$-EA_T outperforms $(\mu + \lambda)$-EA_D significantly.

5 CONCLUSIONS

Constructing point sets of low discrepancy has a prominent role in mathematics and a set of low discrepancy can be seen as being one that is covering the considered space $[0, 1]^d$ in a good way as they

	$(\mu+\lambda)$-EA_C				$(\mu+\lambda)$-EA_D				$(\mu+\lambda)$-EA_T			
	min	mean	std	stat	min	mean	std	stat	min	mean	std	stat
(f1,f4)	0.4836	0.5535	0.0362	$2^{(-)}$,$3^{(-)}$	0.2229	0.2942	0.0512	$1^{(+)}$,$3^{(-)}$	0.2013	0.2354	0.0252	$1^{(+)}$,$2^{(+)}$
(f2, f4)	0.4657	0.5192	0.0256	$2^{(-)}$,$3^{(-)}$	0.3229	0.3708	0.0414	$1^{(+)}$,$3^{(-)}$	0.2816	0.3363	0.0435	$1^{(+)}$,$2^{(+)}$
(f3, f4)	0.5743	0.6296	0.0219	$2^{(-)}$,$3^{(-)}$	0.3590	0.4422	0.0534	$1^{(+)}$	0.3831	0.4113	0.0175	$1^{(+)}$
(f1, f3, f4)	0.7765	0.7997	0.0204	$2^{(-)}$,$3^{(-)}$	0.4303	0.4585	0.0183	$1^{(+)}$	0.4372	0.4604	0.0422	$1^{(+)}$
(f2, f3, f4)	0.7641	0.7962	0.0198	$2^{(-)}$,$3^{(-)}$	0.4197	0.4563	0.0215	$1^{(+)}$	0.3730	0.4514	0.0327	$1^{(+)}$
(f1, f2, f3)	0.7593	0.7836	0.0111	$2^{(-)}$,$3^{(-)}$	0.3900	0.4095	0.0160	$1^{(+)}$	0.3547	0.3988	0.0217	$1^{(+)}$

Table 2: Statistics of discrepancy values for TSP. f1, f2, f3, f4 denote the feature angle_mean, centroid_mean_dist_centroid, nnds_mean, mst_dists_mean respectively.

aim for a good balance of points in every hyper-box with respect to their volume. We have introduced a discrepancy-based evolutionary diversity optimization approach that constructs sets of solutions meeting a given quality criteria and having a low discrepancy with respect to the considered features. Our experimental results for evolving diverse sets of images and TSP instances show that this approach constructs sets of solutions with a much lower discrepancy that the previously used weighted contribution approach according to the given features. Our discrepancy-based diversity optimization process for images makes use of a new random walk mutation operator which reduces the number of required generations to obtain a good diverse set of images by 3 orders of magnitude The best results across all our experimental investigations are obtained by the $(\mu + \lambda)$-EA$_T$, which uses discrepancy-based diversity optimization in conjunction with a tie-breaking rule based on the weighted contribution diversity measure.

Acknowledgment

This work has been supported through Australian Research Council (ARC) grants DP140103400 and DP160102401.

REFERENCES

[1] Bradley Alexander and James Kortman and1 Aneta Neumann. 2017. Evolution of artistic image variants through feature based diversity optimisation. In *Genetic and Evolutionary Computation Conference, GECCO 2017*. 171–178.

[2] David Applegate, William Cook, Sanjeeb Dash, and André Rohe. 2002. Solution of a Min-Max Vehicle Routing Problem. *INFORMS Journal on Computing* 14, 2 (2002), 132–143.

[3] Anne Auger, Mohamed Jebalia, and Olivier Teytaud. [n. d.]. Algorithms (X, sigma, eta): Quasi-random Mutations for Evolution Strategies. In *Proc. of Artificial Evolution EA*.

[4] Gregory W. Corder and Dale I. Foreman. 2009. *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach.* Wiley.

[5] F.-M. De Rainville, C. Gagné, O. Teytaud, and D. Laurendeau. 2012. Evolutionary Optimization of Low-Discrepancy Sequences. *ACM Trans. Model. Comput. Simul.* 22, 2 (2012), 9:1–9:25.

[6] Eelco den Heijer and A. E. Eiben. 2014. Investigating aesthetic measures for unsupervised evolutionary art. *Swarm and Evolutionary Computation* 16 (2014), 52–68.

[7] D. P. Dobkin, D. Eppstein, and D. P. Mitchell. 1996. Computing the discrepancy with applications to supersampling patterns. *ACM Trans. Graph.* 15 (1996), 354–376.

[8] Benjamin Doerr and Carola Doerr. 2015. Optimal Parameter Choices Through Self-Adjustment: Applying the 1/5-th Rule in Discrete Settings. In *Genetic and Evolutionary Computation Conference, GECCO 2015*. ACM, 1335–1342.

[9] C. Doerr, M. Gnewuch, and M. Wahlström. 2014. Calculation of Discrepancy Measures and Applications. In *A Panorama of Discrepancy Theory*. LNCS, Vol. 2107. 621–678.

[10] Carola Doerr and François-Michel De Rainville. 2013. Constructing low star discrepancy point sets with genetic algorithms. In *Proc. of Genetic and Evolutionary Computation Conference, GECCO 2013*. ACM, 789–796.

[11] Wanru Gao, Samadhi Nallaperuma, and Frank Neumann. 2016. Feature-Based Diversity Optimization for Problem Instance Classification. In *Parallel Problem Solving from Nature PPSN 2016*. 869–879.

[12] M. Gnewuch, A. Srivastav, and C. Winzen. 2009. Finding optimal volume subintervals with k points and calculating the star discrepancy are NP-hard problems. *J. Complexity* 25 (2009), 115–127.

[13] M. Gnewuch, M. Wahlström, and C. Winzen. 2012. A new randomized algorithm to approximate the star discrepancy based on threshold accepting. *SIAM J. Numer. Anal.* 50 (2012), 781–807.

[14] Stefan Kern, Sibylle D. Müller, Nikolaus Hansen, Dirk Büche, Jiri Ocenasek, and Petros Koumoutsakos. 2004. Learning probability distributions in continuous evolutionary algorithms - a comparative review. *Natural Computing* 3 (2004), 77–112.

[15] Shuhei Kimura and Koki Matsumura. 2005. Genetic Algorithms Using Low-discrepancy Sequences. In *Proc. of Genetic and Evolutionary Computation Conference, GECCO 2005*. ACM, 1341–1346.

[16] Shuhei KIMURA and Koki MATSUMURA. 2006. Improvement of the Performances of Genetic Algorithms by Using Low-discrepancy Sequences. *Transactions of the Society of Instrument and Control Engineers* 42 (2006), 659–667.

[17] Kresimir Matkovic, László Neumann, Attila Neumann, Thomas Psik, and Werner Purgathofer. 2005. Global Contrast Factor-a New Approach to Image Contrast. *Computational Aesthetics* (2005), 159–168.

[18] J. Matoušek. 2010. *Geometric Discrepancy* (2nd ed.). Springer.

[19] Olaf Mersmann, Bernd Bischl, Heike Trautmann, Markus Wagner, Jakob Bossek, and Frank Neumann. 2013. A novel feature-based approach to characterize algorithm performance for the traveling salesperson problem. *Annals of Mathematics and Artificial Intelligence* 69, 2 (2013), 151–182.

[20] Samadhi Nallaperuma, Markus Wagner, and Frank Neumann. 2015. Analyzing the Effects of Instance Features and Algorithm Parameters for Max-Min Ant System and the Traveling Salesperson Problem. *Front. Robotics and AI* (2015).

[21] Aneta Neumann, Bradley Alexander, and Frank Neumann. 2017. Evolutionary Image Transition Using Random Walks. In *EvoMUSART 2017, Proceedings*. 230–245.

[22] Aneta Neumann, Zygmunt L. Szpak, Wojciech Chojnacki, and Frank Neumann. 2017. Evolutionary image composition using feature covariance matrices. In *Genetic and Evolutionary Computation Conference, GECCO 2017*. ACM, 817–824.

[23] Mark Nixon and Alberto S. Aguado. 2008. *Feature Extraction & Image Processing, Second Edition* (2nd ed.). Academic Press.

[24] R Core Team. 2015. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

[25] Marc Schoenauer, Fabien Teytaud, and Olivier Teytaud. 2011. A Rigorous Runtime Analysis for Quasi-Random Restarts and Decreasing Stepsize. In *Proc. of Artificial Evolution EA 2011*. 37–48.

[26] Kate Smith-Miles, Jano van Hemert, and Xin Yu Lim. [n. d.]. Understanding TSP difficulty by learning from evolved instances.

[27] Olivier Teytaud. 2008. When Does Quasi-random Work?. In *Parallel Problem Solving from Nature, PPSN 2008*. 325–336.

[28] Olivier Teytaud. 2015. Quasi-random Numbers Improve the CMA-ES on the BBOB Testbed. In *Proc. of Artificial Evolution, EA 2015*. 58–70.

[29] Olivier Teytaud and Sylvain Gelly. 2007. DCMA: yet another derandomization in covariance-matrix-adaptation. In *Genetic and Evolutionary Computation Conference, GECCO 2007*. 955–963.

[30] Tamara Ulrich and Lothar Thiele. 2011. Maximizing population diversity in single-objective optimization. In *Genetic and Evolutionary Computation Conference, GECCO 2011*. ACM, 641–648.

[31] M. Wahlström. [n. d.]. ([n. d.]). Implementations of the DEM- and the TA-algorithm. Available at http://www.mpi-inf.mpg.de/~wahl/.