On the use of feature-maps for improved quality-diversity meta-evolution

David M. Bossens
d.m.bossens@soton.ac.uk
University of Southampton
UK

Danesh Tarapore
d.s.tarapore@soton.ac.uk
University of Southampton
UK

ABSTRACT

Quality-Diversity (QD) algorithms evolve a behaviourally diverse archive of high-performing solutions. In QD meta-evolution, one evolves a population of QD algorithms by modifying algorithmic components (e.g., the behaviour space) to optimise an archive-level objective, the meta-fitness. This paper investigates which feature-map is best for defining the behaviour space for an 8-joint robot arm. Meta-evolution with non-linear feature-maps yields a 15-fold meta-fitness improvement over linear feature-maps. On a damage recovery test, archives evolved with non-linear feature-maps outperform traditional MAP-Elites variants.

CCS CONCEPTS

• **Computing methodologies** → *Artificial intelligence*; *Evolutionary robotics*; *Learning paradigms.*

KEYWORDS

quality-diversity algorithms, meta-evolution, representational capacity, evolutionary robotics, damage recovery

ACM Reference Format:

David M. Bossens and Danesh Tarapore. 2021. On the use of feature-maps for improved quality-diversity meta-evolution. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3449726.3459442

1 INTRODUCTION

In *quality-diversity (QD) algorithms* such as MAP-Elites [9], one evolves a large archive of behaviourally diverse and high-performing solutions. An important design choice of a QD algorithm is its behaviour space, the features that define the behavioural diversity across the solutions. Traditionally, behavioural features are chosen by the user, however, complex and non-intuitive features often optimise the intended purpose better. Therefore, an automated approach to the behaviour space may be required.

A promising approach in this regard is QD meta-evolution, in which a meta-level evolutionary algorithm evolves a population of QD-algorithms. Contrasting to automated behaviour spaces [2, 4] and automated operators [5], the meta-level algorithm can modify any component of the QD algorithm and evaluate how well the newly formed QD algorithm performs on a user-defined meta-level objective. This paper explores QD meta-evolution with modifiable behaviour space to allow damage recovery in a robot arm.

2 QUALITY-DIVERSITY META-EVOLUTION WITH FEATURE-MAPS

MAP-Elites (ME) evolves a behaviour-performance map, storing the highest-fitness controllers for each hypercube in a discretised behaviour space [9]. Since ME is not explicitly optimised for generalisation, we use Meta-evolution with CMA-ES [1] to evolve a population of MEs with generalisation as a meta-objective. The system performs the following steps:

(1) Sample new meta-genotypes $\mathbf{W}^1, \dots, \mathbf{W}^\lambda$ from the multivariate normal distribution defined by CMA-ES.
(2) For $i \in \{1, \dots, \lambda\}$, use meta-genotype $\mathbf{W}^i$ to construct a new map $\mathcal{M}^i$ based on existing solutions in the database.
(3) For $i \in \{1, \dots, \lambda\}$, ME($\mathbf{W}^i$) further evolves $\mathcal{M}^i$ and all newly generated solutions are stored in the database. In this paper, $\mathbf{W}^i$ parametrises the behaviour space only.
(4) Evaluate each meta-genotype $i \in \{1, \dots, \lambda\}$ on the meta-fitness $\mathcal{F}(\mathbf{W}^i)$ (see Section 3 for its definition).
(5) CMA-ES updates the mean, covariance, and step size , applying the $(\mu/\mu_W, \lambda)$-CMA Evolution Strategy [6].

The system learns how to learn by modifying the behaviour space to optimise meta-fitness, hereby determining the ME's initial behaviour-performance map (step 2) and how its solutions are selected for reproduction (step 3). At the end of the run, the system returns the behaviour-performance map with the highest meta-fitness, which can then be used in the application of interest.

To define the behaviour space based on the meta-genotype, we generalise the purely linear transformation used in earlier work [1] to *feature-maps.* Feature-maps transform base-features $\mathbf{b} \in [0, 1]^{N_b}$, a large number of behavioural features handcrafted by the user, to a low-dimensional behavioural descriptor $\beta \in [0, 1]^D$. To prevent loss of quality-diversity, the database is not a circular buffer as in prior work but instead it stores the highest-performing solutions for each coarse-grained bin in the base-behavioural space (see Section S1 in Supplemental Materials).

The study includes three feature-maps. First, the **linear** transformation used in earlier work [1] is included:

$$\beta = \mathbf{W}\mathbf{b}\,, \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{D \times N_b}$. An expanding normalisation is added to increase the coverage of the behaviour space. Second, to demonstrate the need to combine features, a **feature-selector** is included,

$$\beta = b_{j^1}, \dots, b_{j^D}\,, \tag{2}$$

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07.
https://doi.org/10.1145/3449726.3459442

where $j^i = \arg\max_{j \in N_b} W_{ij}$ selects for each feature $i \in \{1, \dots, D\}$ the base-feature b_j with highest W_{ij}. Third, to demonstrate the need for non-trivial feature-maps, we include a **non-linear** transformation using a neural network with a single hidden layer,

$$\begin{aligned} h(\mathbf{W}, \mathbf{b}) &= \mathbf{W}^2 S_{N_b}(\mathbf{W}^1 \mathbf{b} + B^1) + B^2 \\ \beta &= S_{N_h}(h(\mathbf{W}, \mathbf{b})), \end{aligned} \tag{3}$$

where $S_N(\mathbf{x}) = 1/(1 + \exp(-\alpha_s \mathbf{x}/(N+1)))$ is an elementwise sigmoid function correcting for the scale of the weighted sum of a number of N incoming activations; α_s is an empirically defined scaling factor; and now $\mathbf{W}$ is composed of a weight matrix from input to hidden layer, $\mathbf{W}^1 \in \mathbb{R}^{N_h \times N_b}$, a weight matrix from hidden layer to output, $\mathbf{W}^2 \in \mathbb{R}^{D \times N_h}$, and the corresponding bias units $B^1, B^2 \in \mathbb{R}$. Universal approximation theorems [7, 8] imply that such networks can represent all multi-variate functions over bounded intervals, assuming a sufficient number of neurons.

3 EXPERIMENT SETUP

We set up 8-joint robot arm experiments that were part of the seminal study of Cully et al. [3]. Each of the robot's eight joints can rotate in $[-\pi/2, \pi/2]$rad to position its gripper within a semi-circle span. The genotype, $\mathbf{g}$, represents the desired angles for the joints. The fitness $f(\mathbf{g})$ is the negative variance of the angles, discouraging zigzag motions for energy-efficiency and distributing movement equally among the joints for robustness to failures.

The meta-fitness $\mathcal{F}$ aims for the robot arm to uniformly cover in as fine-grained manner as possible its entire semi-circle span regardless of any damage. It is evaluated by taking 10% of the solutions from the behaviour-performance map and then computing the summed pairwise distance between the gripper positions. This metric is then averaged across 16 damages in the damage set $\mathcal{D}$, two angles chosen for each of the eight joints. The use of one positive and one negative angle randomly chosen in the range of the joint avoids bias to a particular orientation and reduces the variance of meta-fitness evaluations over time.

The experiments consider 5 behaviour spaces. **Position (2D)** and **Polar (2D)** use the Cartesian and polar coordinates, respectively. **JointPairAngle (4D)** is the angle in spanned by connecting joint $i-2$ to joint i for all $i \in \{2, 4, 6, 8\}$. **AngleSum (6D)** is the average value for each consecutive triplet of bottom-level genes. Meta-evolution (**Meta (4D)**) concatenates the above into 14 base-features and maps them onto a 4D feature space.

Experimental parameters are found in Supplemental Materials.

4 RESULTS

The meta-fitness is strongly dependent on the feature-map (see Figure 1). Linear and feature-selection feature-maps improve rapidly in meta-fitness early on but stagnate on a plateau for the rest of meta-evolution. Non-linear feature-maps start slowly but improve rapidly between 10 and 20 million function evaluations. They reach the highest meta-fitness of around 15 000 m, representing near-uniform spread for 300 solutions (10% of around 3000 solutions), and provide a 6-fold improvement over feature-selection and a 15-fold improvement over linear feature-maps. This illustrates the trade-off of high-complexity functions: non-linear feature-maps can represent any function but they require more data. Linear feature-maps yield low meta-fitness because they often output similar target-features for diverse combinations of base-features.

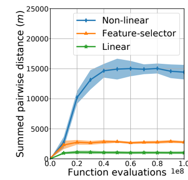
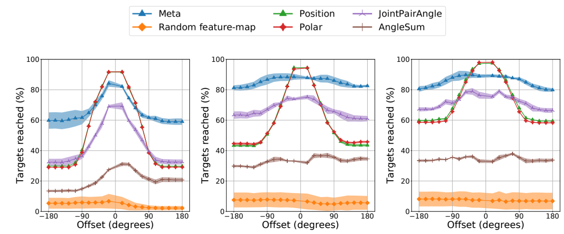


Figure 1: Effect of feature-map on meta-fitness (y-axis; Mean ± SD over 5 replicates) across function evaluations (x-axis).



(a) Joint 1 (b) Joint 5 (c) Joint 8

Figure 2: Test on unseen damages. For each offset (x-axis), the coverage of the semi-circle span of the robot is shown (y-axis; Mean ± SD across 5 replicates). For Meta, the behaviour-performance map is formed from CMA-ES' mean.

In a test after evolution, individual joints $i \in \{1, \dots, 8\}$ are damaged by offsets in $\epsilon \in \{-1, -0.9, \dots, -0.1, 0.1, \dots, 0.9, 1\}\pi$ rad, resulting in an angle $\theta_i \leftarrow \max(-\pi/2, \min(\pi/2, \theta_i + \epsilon))$rad. Meta-evolution with non-linear feature-map yields the highest coverage for high-severity damages and otherwise comparable coverage to Polar and Position (see Figure 2). Therefore, any target position chosen by the user would likely be reachable after damage.

ACKNOWLEDGEMENTS

This work was supported by EPSRC under the New Investigator Award grant, EP/R030073/1 (Tarapore). The authors acknowledge the IRIDIS High Performance Computing Facility and thank Arnold Benedict for initial work on the simulator.

REFERENCES

[1] David M. Bossens, Jean-Baptiste Mouret, and Danesh Tarapore. 2020. Learning behaviour-performance maps with meta-evolution. In *GECCO'20 - Genetic and Evolutionary Computation Conference*. Cancun, Mexico, 49–57.
[2] Antoine Cully. 2019. Autonomous skill discovery with quality-diversity and unsupervised descriptors. In *Proceedings of the Genetic and Evolutionary Computation Conference, (GECCO '19)*. Prague, Czech Republic, 81–89.
[3] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. 2015. Robots that can adapt like animals. *Nature* 521, 7553 (may 2015), 503–507.
[4] Antoine Cully and Yiannis Demiris. 2018. Hierarchical Behavioral Repertoires with Unsupervised Descriptors. In *GECCO '18: Genetic and Evolutionary Computation Conference*. 69–76. https://doi.org/10.1145/3205455.3205571 arXiv:1804.07127
[5] Adam Gaier, Alexander Asteroth, and Jean-Baptiste Mouret. 2020. Discovering Representations for Black-box Optimization. In *GECCO'20 - Genetic and Evolutionary Computation Conference*. Cancun, Mexico.
[6] Nikolaus Hansen. 2016. The CMA Evolution Strategy: A Tutorial. *arXiv preprint arXiv:1604.00772* (2016), 1–39. arXiv:1604.00772 http://arxiv.org/abs/1604.00772
[7] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. 1989. Multilayer feedforward networks are universal approximators. *Neural Networks* 2, 5 (1989), 359–366.
[8] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. 1993. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks* 6, 6 (1993), 861–867.
[9] Jean-Baptiste Mouret and Jeff Clune. 2015. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909v1* (2015), 1–15.