

Evolving Gaits for Damage Control in a Hexapod Robot

Christopher Mailer
mlrchr001@myuct.ac.za
University of Cape Town
South Africa

Geoff Nitschke
gnitschke@cs.uct.ac.za
University of Cape Town
South Africa

Leanne Raw
leanne.raw@uct.ac.za
University of Cape Town
South Africa

ABSTRACT

Autonomous robots are increasingly used in remote and hazardous environments, where damage to sensory-actuator systems cannot be easily repaired. Such robots must therefore have controllers that continue to function effectively given unexpected malfunctions and damage to robot morphology. This study applies the *Intelligent Trial and Error* (IT&E) algorithm to adapt hexapod robot control to various leg failures and demonstrates the IT&E *map-size* parameter as a critical parameter in influencing IT&E adaptive task performance. We evaluate robot adaptation for multiple leg failures on two different map-sizes in simulation and validate evolved controllers on a physical hexapod robot. Results demonstrate a trade-off between adapted gait speed and adaptation duration, dependent on adaptation task complexity (leg damage incurred), where map-size is crucial for generating behavioural diversity required for adaptation.

CCS CONCEPTS

• **Computing methodologies** → **Evolutionary robotics.**

KEYWORDS

Evolutionary Robotics, Quality-Diversity Algorithms, Bayesian optimisation, Damage recovery, Behavioural diversity

ACM Reference Format:

Christopher Mailer, Geoff Nitschke, and Leanne Raw. 2021. Evolving Gaits for Damage Control in a Hexapod Robot. In *2021 Genetic and Evolutionary Computation Conference (GECCO '21)*, July 10–14, 2021, Lille, France. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3449639.3459271>

1 INTRODUCTION

Autonomous robots will potentially explore distant or hostile environments which would otherwise be entirely inaccessible, or pose a substantial risk to humans [1, 4]. This provides enormous benefits to scientific exploration [2, 3], search and rescue [11], and disaster recovery [24]. Due to the complexity and unpredictability of these harsh environments, hardware reliability has been identified as a significant challenge [5]. This is particularly pertinent as *in situ* repair or retrieval is not possible. Humans and animals have evolved the ability to rapidly adapt to unexpected injuries and impediments. Robots with this ability could continue to function even when faced with unexpected hardware failures or damage. Unlike traditional

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
GECCO '21, July 10–14, 2021, Lille, France

© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8350-9/21/07...\$15.00
<https://doi.org/10.1145/3449639.3459271>

Algorithm 1 IT&E adapted from [9]

```
1: Initialize:  
   Behaviour-performance map with MAP-Elites  
2: while in mission do  
3:   if significant performance drop then           ▶ Leg failure  
4:     Adaptation with M-BOA
```

diagnosis and contingency methods [18], the ability to adapt does not require anticipation and planning of possible failures [25, 26].

Recently model-based reinforcement learning approaches have shown promise for legged robot adaptation tasks. These involve learning a gait-dynamics model using a simulation of the robot and reinforcement learning, and then optimising the learnt policy [6, 19, 23, 28] using state data from sensors on the real robot. An issue with such methods is their use of high dimensional state information relative to the action space. For example, a robot with an 18D action space required 48D of state information. This is infeasible for many applications given strict limits on state information from the robot's *Inertial Measurement Unit* (IMU)¹. The *Intelligent Trial and Error* (IT&E) algorithm [9], (algorithm 1), takes a different approach to adaptation. First a diverse collection of gaits (*behaviour-performance map*) is generated using the MAP-Elites algorithm [22]. If the current gait is not performing as expected then the behaviour-performance map is explored online using Bayesian optimisation [16] to find a gait which performs optimally with the failure.

IT&E was tested on an 18 degree-of-freedom hexapod robot (*Pexod*), enabling it to adapt to various leg failures in under two minutes using only 1D feedback about its walking speed. This low dimensional feedback requirement makes IT&E ideal for implementation on robots with limited sensors. *Pexod* [9] used a simple open-loop oscillator as the gait controller for locomotion. While this adequately highlighted the efficacy of the IT&E algorithm, its simplicity severely limits its applications to legged robots which require precise control over the foot trajectories and placement.

This study however uses a legged hexapod robot with a kinematic and trajectory based gait controller where the feet follow a designed trajectory and can use body orientation data from the IMU for stabilisation [7]. Our hexapod was developed for potential use in search and rescue scenarios [7], but currently has no means of detecting the failure of its actuators, and due to its fixed tripod gait, is rendered immobile if even only one of its 18 actuators fail. This is unnecessary as the robot is over-actuated and therefore inherently redundant. Even with the loss of an entire leg there are still five fully functional legs remaining to continue with walking. As a case study, we thus extend IT&E to adapt our hexapod gait behaviour *online*, in response to leg damage.

¹The hexapod used in this study only provides 9D state information from its IMU

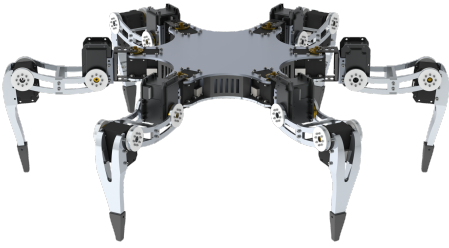


Figure 1: Rendering of the hexapod robot

1.1 Research Objectives and Contribution

The study objective is to present new methods which integrate IT&E with our hexapod’s trajectory based gait controller and evaluate task performance as adaptation to various leg failures in both simulation and on the physical robot. Study results indicate that the behaviour-performance map is a crucial component to adaptation in IT&E with various characteristics significantly impacting adaptive task performance. One such characteristic is the map size. Previous work has investigated the impact of map size selection, however this analysis was limited to the performance of the behaviours generated within the map [31]. A key contribution of this study is thus elucidation of the impact of different map sizes on the evolution of gaits (suitable adaptation of task performance) given various degrees of morphological (leg) damage.

2 PHYSICAL ROBOT

Experiments used a custom hexapod robot (figure 1), with a circular symmetrical design, 18 degrees-of-freedom and a weight of approximately 3.5 kg. Each of its six legs are actuated by two Dynamixel RX-28 servos [27] at the coxa and tibia joints and one Dynamixel RX-64 servo [27] at the femur joint. The control algorithms run on an on-board STM32F407VGT6 micro-controller [29], and the robot is powered by either the built-in batteries or a 25 V external power supply. The primary sensor on the robot is an Xsens MTi-28A33G35 IMU [33] which provides 9D body inertial state information.

2.1 Gait Controller

The hexapod robot uses the closed-loop kinematic gait controller from [7] to walk. High-level gait commands sent by the user are transformed into foot trajectories and then servo motions. A straight line trajectory is used for the support phase foot motion and a 6th order polynomial is used for the swing phase foot motion. The foot trajectories are transformed into joint angles using inverse-kinematics which are then sent to the servos at 120 Hz. The controller for this study was used in its open-loop configuration with base stabilisation turned off. Controller parameterisation and leg trajectories were inspired by related work [32] and included fundamental components of statically stable legged gaits such as support polygon [20] and foot timing [17], thus producing diverse symmetrical and asymmetrical gaits and common hexapedal gaits [13].

Leg Radius (r_i) Controls how close the foot support phase is to the body during walking.

Leg Angle (θ_i) Controls hip angle of a leg during walking. Angle is relative to default radially outwards leg position.

Table 1: Gait parameter range

Parameter	Symbol	Range
Body Velocity	v	0 – 0.5 m s ⁻¹
Body Height	h	0 – 0.2 m
Leg Radius	r_i	0 – 0.3 m
Leg Angle	θ_i	-100 – 100 °
Leg Step Height	s_i	0.05 – 0.2 m
Leg Phase Offset	ϕ_i	0 – 180 °
Leg Duty Factor	β_i	0 – 1.0

Leg Step Height (s_i) Controls the maximum height attained by the foot during the swing phase.

Leg Duty Factor (β_i) Controls the portion of time a leg spends in the support phase out of the whole gait cycle.

Leg Phase Offset (ϕ_i) Controls the phase offset of a legs support and swing cycle.

The i subscript denotes that each leg has a unique parameter. The next two parameters are however not unique for each leg and serve to impose a degree of coordination between each of the feet.

Body Velocity (v) Common velocity of each foot driving the base in the commanded direction during their support phase.

Body Height (h) Determines the common height the feet will be at during the support phase to maintain a level base.

This amounts to a total of 32 parameters for control of the gait with 5 parameters per leg and 2 whole-body parameters. The leg cycle was kept fixed for each leg at 1 Hz to stop the leg cycles moving in and out of phase. The default gait used on the robot is the tripod gait [13] facilitating greater speed [7]. This gait has the following parameters and was used as the reference gait in our experiments:

$$\begin{aligned}
 v &= 0.3 \text{ m s}^{-1} & r_i &= 0.15 \text{ m} & \theta_i &= 0^\circ & \phi_{1,3,5} &= 0^\circ \\
 h &= 0.14 \text{ m} & s_i &= 0.05 \text{ m} & \beta &= 0.5 & \phi_{2,4,6} &= 180^\circ
 \end{aligned}$$

The bounds of the gait controller parameters are determined by the reachable bounds of the legs, and the range for these parameters is shown in table 1. The lower bound of the step height was set to 5 cm to ensure the feet were lifted for the swing phase. If a foot is commanded outside of its reachable area due to the controller parameters, the controller throws an error.

3 SIMULATOR

Our dynamics simulator uses the PyBullet [8] physics engine running at 240 Hz, with a URDF model of the hexapod and MAP-Elites to generate behaviour-performance maps. The URDF model was created from a SolidWorks model of the robot with almost identical dimensions and inertial properties. The inertial properties for the links were calculated using SolidWorks and the geometry of the links was modelled with STL files. Simplified geometries with the same outer dimensions were used to reduce the model loading time.

3.1 Approximations

The robot interacts with the environment through contacts between the feet and the ground. Simulating foot contacts is challenging as the friction coefficient can vary substantially between surfaces.

We chose a relatively high friction coefficient of 0.7 to disincentivise dragging of the feet as this was an undesirable behaviour for the physical robot. The gait controller on the physical robot was replicated with the only difference being that the controller in simulation runs at 240 Hz while the controller on the physical robot runs at 120 Hz. The physical servo torque and angular velocity limits were included in the simulation, however the torque-speed relationship was ignored.

4 MAP GENERATION

The first stage of the IT&E adaptation algorithm is the generation of a collection of diverse and high performing gaits, referred to as a behaviour-performance map, which serves as experience the robot can draw from when it needs to adapt [9]. The map² is generated with the CVT-MAP-Elites algorithm presented in [31] and reproduced in algorithm 2. We used the Python implementation provided in the `pymap_elites` GitHub repository [21]. CVT-MAP-

Algorithm 2 CVT-MAP-Elites adapted from [31]

```

1:  $C \leftarrow \text{CVT}(k)$ 
2:  $(\mathcal{X}, \mathcal{P}) \leftarrow \text{create\_empty\_archive}(k)$ 
3: for  $i = 1 \rightarrow G$  do
4:    $\mathbf{x} \leftarrow \text{random\_solution}()$ 
5:    $\text{ADD\_TO\_ARCHIVE}(\mathbf{x}, \mathcal{X}, \mathcal{P})$ 
6:   for  $i = 1 \rightarrow I$  do
7:      $\mathbf{x}_1, \mathbf{x}_2 \leftarrow \text{random\_selection}(\mathcal{X})$ 
8:      $\mathbf{x}' \leftarrow \text{SBX}(\mathbf{x}_1, \mathbf{x}_2)$ 
9:      $\text{ADD\_TO\_ARCHIVE}(\mathbf{x}', \mathcal{X}, \mathcal{P})$ 
10:  return  $(\mathcal{X}, \mathcal{P})$ 
11: function  $\text{ADD\_TO\_ARCHIVE}(\mathbf{x}, \mathcal{X}, \mathcal{P})$ 
12:    $p, \mathbf{b} \leftarrow \text{simulate}(\mathbf{x})$ 
13:    $c \leftarrow \text{index\_of\_closest\_centroid}(\mathbf{b}, C)$ 
14:   if  $\mathcal{P}(c) = \emptyset$  or  $\mathcal{P}(c) < p$  then
15:      $\mathcal{X}(c) \leftarrow \mathbf{x}, \mathcal{P}(c) \leftarrow p$ 

```

Elites begins by discretising a user-defined behaviour space into k evenly spaced niches with a Centroidal Voronoi tessellation (CVT) to ensure a uniform distribution of behaviours. Another version of MAP-Elites [22] uses a grid-based approach to discretisation, however this does not provide required precise control over the number of niches. Once k niches have been created, the behaviour-performance map $(\mathcal{X}, \mathcal{P})$ is initialised to store solutions (\mathcal{X}) and their corresponding performances (\mathcal{P}) . To provide an initial population, G random solutions (\mathbf{x}) are initialised and stored in the map. A solution represents the 32 gait controller parameters in the form:

$$\mathbf{x} = [\underbrace{v \quad h}_{\text{body}} \quad \underbrace{r_1 \quad \theta_1 \quad s_1 \quad \beta_1 \quad \phi_1}_{\text{leg 1}} \quad \dots \quad \underbrace{r_6 \quad \dots \quad \phi_6}_{\text{leg 6}}]$$

The parameters and gait controller are evaluated with our simulation of the hexapod to determine a performance (p) and a behaviour descriptor (\mathbf{b}) associated with the gait. The behaviour descriptor determines the location of the solution in the behaviour space and therefore which niche it will occupy. After random initialisation MAP-Elites follows a parent selection and genetic variation loop

²The term behaviour-performance map is used interchangeably with map

[14]. Solutions are randomly selected from the map and varied with the *Simulated Binary Crossover* (SBX) operator [12]. The mutated solution (\mathbf{x}') is added to the map in its corresponding niche provided the niche is empty $(\mathcal{P}(c) = \emptyset)$ or the present solution is lower performing $(\mathcal{P}(c) < p)$. The selection and variation loop runs for I evaluations gradually filling the map with behaviourally diverse and high-performing solutions. We ran MAP-Elites for 40 million evaluations with the following parameters (appendix A.1). The number of evaluations was guided by related work [9] and job time limits on the computing cluster. Due to the stochastic nature of the algorithm the final collection of gaits can vary with each run, so 20 independent maps were generated to account for this.

4.1 Performance Metric

Gait performance (p) was defined as the average velocity of the center of the hexapod's body along a single axis in the simulation. It is worth noting that this gait performance relates directly to the velocity parameter (v) in the gait controller, however this parameter cannot simply be increased to produce faster gaits as the legs will begin to collide. Full coordination of the legs therefore needs to be evolved to produce faster walks. To disincentivise unsafe gaits the following conditions resulted in immediate termination of the simulation with the gait being assigned a performance of 0 m s^{-1} :

- Collisions between the legs
- Collisions between the base and the ground
- Leg kinematic singularities

The velocity along a single axis incentivised the generation of gaits which exhibited straight line motion in a desired direction. The robot was oriented with three legs on either side of this desired walking direction (figure 3a).

4.2 Behaviour Descriptor

The behaviour descriptor (\mathbf{b}) to describe gaits was a 6-dimensional vector describing time portions each foot spends in contact with the ground (*duty factor* [17]). This is a common metric used to describe gaits and is expressed mathematically as:

$$\mathbf{b} = [b_1 \quad \dots \quad b_6] = \left[\frac{\sum c_1(t)}{n_t} \quad \dots \quad \frac{\sum c_6(t)}{n_t} \right] \quad (1)$$

Where $c_i(t)$ represents the i^{th} Boolean foot-ground contact at time-step t and n_t is the number of time-steps. Contacts are computed by the physics engine per simulation time-step. This descriptor was also demonstrated to perform well in 6-legged robot adaptation [9].

4.3 Map Size Experiments

Behaviour-performance map size is determined by the number of niches (k) which ultimately controls the degree of diversity. While the behaviour space remains unchanged, a map with more niches will contain a greater number of novel gait solutions within this space and therefore exhibit a greater degree of diversity. Solution diversity was identified as a crucial component in enabling adaptation as it improved the chances of finding a compensatory gait [9]. However, greater diversity comes at the cost of reduced selective pressure for performance [31], requiring more evaluations and computational resources to achieve similar performance for larger maps. Clearly sacrificing some map diversity will result in greater

walking performance, however, it is unclear how this will translate to adaptation task performance where diversity also plays a role. To investigate this we generated two different map sizes (20k and 40k), for comparison. These sizes were selected based on previous implementations [9, 22, 30, 31] and prior experimentation.

4.4 Computing Hardware

Map generation was executed on the South African Lengau high-performance computing cluster using 240 2.6 GHz Intel Xeon cores across 10 compute nodes. Concurrent evaluations was achieved with the python mpi4py library [10]. To reduce MPI communication overhead and overall runtime we used a batch size of 2390 so each of the 239 worker processes executes a batch of 10 simulations before returning results to the master process. Generation of a single map took on average 40 hours with a peak cumulative RAM usage of approximately 150 GB and an average core utilisation of 40%.

4.5 Quality Metrics

To compare the 20k and 40k behaviour-performance maps we used the same 4 metrics from [22, 30] to quantify the quality of the maps. The quality metrics are the global performance, global reliability, precision, and coverage. The global performance represents the proximity to the highest performing solution across all runs. Map reliability is the average performance of the niches filled during any run while precision is only the average performance across niches filled for that particular run. Coverage is simply the number of niches filled for a single run. In addition to these metrics we also considered the maximum performance, average performance, and coverage of the maps during generation.

5 ADAPTATION

When a failure occurs on the robot, gaits from the map no longer perform as expected as they are all evolved with a fully functional robot. As a result of the diversity of gaits within the map, some of the gaits will perform well with the failure, while others will perform worse. Adaptation involves intelligently exploring the map to find a gait which performs well. This is achieved using the *Map-based Bayesian Optimisation Algorithm* (M-BOA) presented in [9] and reproduced in algorithm 3. A prediction of how the gaits in the map will perform (P) is modelled with a Gaussian process (\mathcal{N}), using the expected performances from the map (\mathcal{P}) as a prior, similar to traditional Bayesian optimisation [16]. The gait predicted to perform the best is selected based on the *upper confidence bound* (UCB) acquisition function. This gait is trialed on the failed robot and its performance (p_{t+1}) is used to update the prediction model. This process repeats building a better prediction as to how gaits in the map will perform with the failure, and terminates once the actual performance (p_{t+1}) is within α of the maximum predicted performance. The parameters for this algorithm were kept the same as [9] and can be found in appendix A.2. The primary metrics for evaluating adaptation are the speed of the final gait and the number of trials required for adaptation, analogous to the duration for adaptation. We evaluated adaptation for these metrics in simulation and reality with each gait trial lasting 5 s, as in related work [9].

Algorithm 3 M-BOA adapted from [9]

```

1:  $\forall \mathbf{x} \in \text{map}$ 
2:  $P(f(\mathbf{x})|\mathbf{x}) = \mathcal{N}(\mu_0(\mathbf{x}), \sigma_0^2(\mathbf{x}))$ 
3:  $\mu_0(\mathbf{x}) = \mathcal{P}(\mathbf{x})$ 
4:  $\sigma_0^2(\mathbf{x}) = k(\mathbf{x}, \mathbf{x})$ 
5: while  $\max(\mathbf{P}_{1:t}) < \alpha \max(\mu_t(\mathbf{x}))$  do
6:    $\mathbf{x}_{t+1} \leftarrow \text{argmax}_{\mathbf{x}}(\mu_t(\mathbf{x}) + \kappa \sigma_t(\mathbf{x}))$ 
7:    $p_{t+1} \leftarrow \text{evaluate}(\mathbf{x}_{t+1})$ 
8:    $P(f(\mathbf{x})|\mathbf{P}_{1:t+1}, \mathbf{x}) = \mathcal{N}(\mu_{t+1}(\mathbf{x}), \sigma_{t+1}^2(\mathbf{x}))$ 
9:    $\mu_{t+1}(\mathbf{x}) = \mathcal{P}(\mathbf{x}) + \mathbf{k}^T \mathbf{K}^{-1} (\mathbf{P}_{1:t+1} - \mathcal{P}(\mathcal{X}_{1:t+1}))$ 
10:   $\sigma_{t+1}^2(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}^T \mathbf{K}^{-1} \mathbf{k}$ 
11:  where:
12:   $\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \dots & k(\mathbf{x}_1, \mathbf{x}_{t+1}) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_{t+1}, \mathbf{x}_1) & \dots & k(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}) \end{bmatrix}$ 
13:   $\mathbf{k} = [k(\mathbf{x}, \mathbf{x}_1) \quad k(\mathbf{x}, \mathbf{x}_2) \quad \dots \quad k(\mathbf{x}, \mathbf{x}_{t+1})]$ 

```

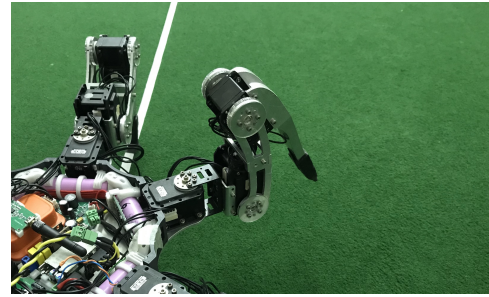


Figure 2: Leg locked in the failed position

5.1 Failure Scenarios

To test adaptation, we defined four failure scenarios. These encompass all configurations for one and two full leg failures. Failures of more than two legs were not considered as this would require dynamic stabilisation to remain balanced while walking, which is beyond gait controller capabilities. The failure scenarios are:

- S1 One failed leg
- S2 Two failed legs separated by two functional legs
- S3 Two failed legs separated by one functional leg
- S4 Two adjacent failed legs

Each failure scenario can occur in a number of orientations around the body. Scenario's S1, S3, and S4 have 6 unique orientations, and S2 has 3 unique orientations, resulting a total of 21 failure combinations. To approximate a failure, rather than removing the legs as in [9], the leg was locked in a retracted position shown in figure 2. This ensures that the leg cannot provide support and will not interfere with the other legs whilst still retaining its dead-weight, analogous to how an animal might lift up an injured leg.

5.2 Simulated Experiments

The aim of the simulated experiments was to evaluate the influence of the different map sizes on overall adaptation performance without the additional variables associated with transferring controllers to the real robot. Both of the 20k and 40k behaviour-performance

Table 2: Physical adaptation experiment configuration

Experiment	Failure Scenario	Map	
1	None	40k_1	
2	None	40k_2	
3	S1	Leg 1	40k_2
4		Leg 4	40k_2
5	S2	Leg 1 & 4	40k_4
6		Leg 1 & 4	40k_6

map sizes were tested with each of the 21 combinations of failure scenarios and orientations, equating to 840 independent adaptation tests. Prior work showed that maps of this size would be adequate for a variety of hexapod leg adaptation tasks [9, 22, 30, 31], however, there is not a clear method for selecting adequate map sizes. The same performance metric introduced in section 4.1 was used as the gait performance feedback (p_{t+1}) for M-BOA; however, the early termination conditions were ignored. Other than these changes the simulator was kept identical to one used in the map generation stage. The performance of the reference gait without adaptation was also evaluated with each of the failures for comparison.

5.3 Physical Experiments

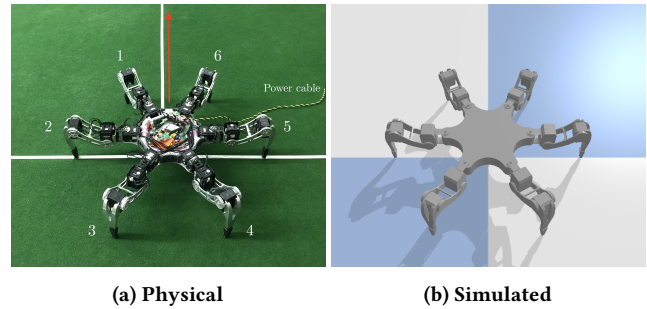
We performed a small number of preliminary experiments with available maps to test adaptation performance on the real robot. The goal of these experiments was to determine whether the evolved gait controllers could be successfully transferred to the real robot, and whether M-BOA resulted in adaptation to leg failures. The case with no leg failures was used to investigate the controller transfer, and failure scenarios S1 and S2 were used to investigate adaptation performance. For S1 and S2 we only considered the orientations with legs 1 and 4. There was no logic behind this decision as in reality any leg could fail. At the time of testing only 10 of the 40k maps were available, so we used the four of these maps with the highest mean performance across all niches (maps 1, 2, 4 and 6). These maps and failures were combined into the following experiment configurations (table 2).

The robot in the physical environment shown in figure 3a was configured identically to the simulated environment shown in figure 3b. For each trial the gait performance was manually measured and fed back into the M-BOA algorithm. A notable difference between the simulated and physical environment is the ground surface. The simulation had perfectly smooth plane with constant friction while the physical environment had polyester carpeting. Otherwise the two environments could be considered comparable.

6 RESULTS AND DISCUSSION

MAP-Elites was run for 40M evaluations for both the 20k and 40k maps, and 20 independent runs were done of each map size resulting in a total of 40 maps. The selection of 40M evaluations as the termination criterion was due to cluster compute time limits.

Figure 4 shows the gait performance and niche coverage progress of these maps. After approximately 10M evaluations the majority of the niches (85%) have been filled and the map coverage shown

**Figure 3: Hexapod robot aligned with the coordinate axes.**

in figure 4c remains relatively stable. For both map sizes 40M evaluations appears to have evolved sufficiently high performing gaits. The smaller 20k map consistently has higher maximum and mean gait speeds shown in figure 4a and figure 4b. This is consistent with results from [31] and attributed to greater evolutionary pressure for performance given fewer niches. With this added performance pressure the maximum performance in the smaller maps reaches the maximum gait speed and plateau's at 0.5 m s^{-1} which is governed by the body velocity parameter (v) range in table 1. While MAP-Elites is a stochastic algorithm, the shaded region in figure 4 shows that the 20 independent runs remain reasonably consistent for the two map sizes. Overall these results show that MAP-Elites is performing as expected and 40M evaluations is adequate.

Final map quality is shown in figure 5 based on the four metrics mentioned in section 4.5. Added performance pressure as a result of the smaller map is confirmed when looking at the quality metrics in figure 5. The smaller 20k map has a significantly higher ($p < 3 \times 10^{-18}$, Welch's t-test [15]) global performance (figure 5a).

Reliability and precision is similar for both map sizes indicating that even with the performance pressure disparity, both map sizes contain on average solutions which perform within 80% of the highest performing solution in each niche. Both map sizes also have similar coverage of about 90%, indicating that the gait controller presented here exhibits sufficient diversity to fill the behavioural space. Furthermore, the number of niches only changes the degree of discretisation of the behaviour space and therefore does not alter the final map coverage. To summarise the map generation results, the smaller map contains higher performing gaits as the focus is less on diversity and more on performance compared to the larger map given identical computing resources. Additionally our proposed parameterised gait controller is both evolvable and exhibits sufficient behavioural diversity to adequately fill the behaviour space.

6.1 Simulation

With the IT&E algorithm the simulated hexapod consistently walks significantly faster ($p < 4 \times 10^{-23}$, Welch's t-test [15]) than the reference gait after adapting to a leg failure for both map sizes, as shown in figure 6a. While there is a substantial drop in performance of 0.2 m s^{-1} for the reference tripod gait with leg failures there is little to no drop in performance after adaptation. These adaptation results are consistent with the results from [9] and show that the

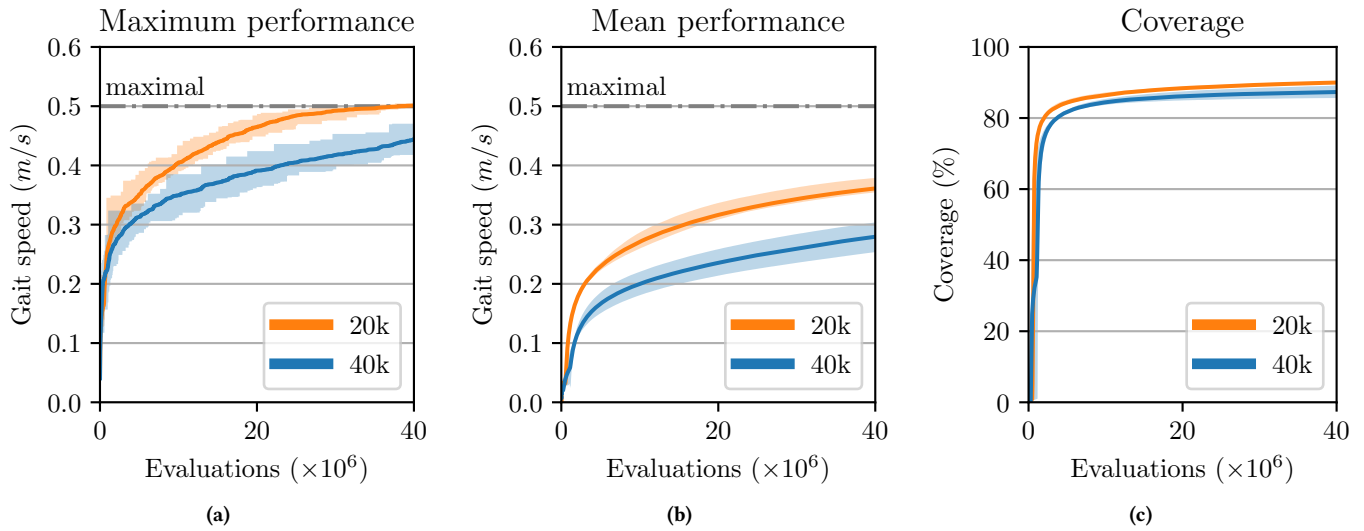


Figure 4: Performance and coverage of behaviour-performance maps (a) The gait speed of the highest performing gait in the map (b) Mean performance of all of the gaits in the map (c) Percentage of filled niches in the map. For all figures, solid lines represent the means over 20 independently generated maps and shaded regions extend to maximum and minimum range.

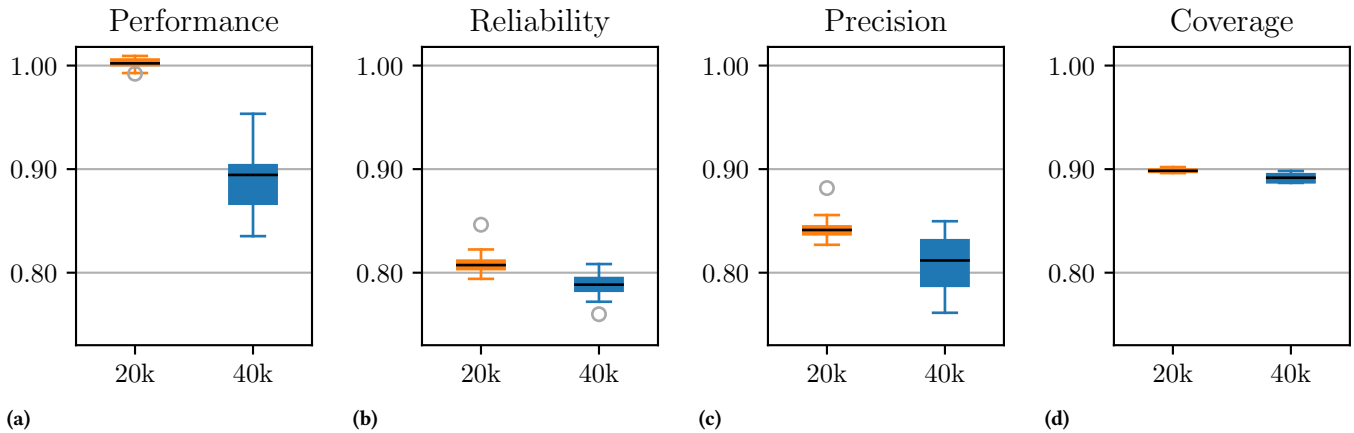


Figure 5: Behaviour-performance map quality across all (normalised) quality metrics (a) Map global performance (proximity to highest-performing solution). (b) Map global reliability (average performance of fill-able niches). (c) Map precision (average performance of niches filled). (d) Map coverage (number of niches filled). Each box plot shows distribution across 20 independently generated maps for two map sizes. Whiskers extend to 25th and 75th percentiles. Outliers shown with grey circles.

M-BOA algorithm is performing as expected. Failure scenarios S2–S4 all have 2 failed legs, however adapted performance gradually decreases from S2 to S4 indicating an increased difficulty. This is attributed to weight imbalance as a result of failed legs being closer together, and the limited ability of the functional legs to move to compensate for this as the gaits in the map are trained using a fully functional hexapod and are penalised when the legs collide.

This results in the evolved gaits still having the legs evenly spaced around the body as if there were 6 functional legs even though there are now only 4. When considering the impact of map size on adaptation, figure 6a shows that the smaller performance oriented 20k map results in a significantly faster ($p < 9 \times 10^{-17}$,

Welch’s t-test [15]) gaits for failure scenarios S1–S3. Furthermore, this is achieved in a similar number of trials as the larger map shown by figure 6b. The increased focus on performance has transferred directly to adaptation resulting in a greater performance after adaptation. Also, halving diversity for the smaller map required to achieve this higher performance does not appear to have detrimentally impacted either adaptation task performance metric.

For the most difficult failure scenario S4 the larger map overall performs better as it is able to achieve similar adapted performance in significantly fewer ($p < 4 \times 10^{-7}$, Welch’s t-test [15]) trials than the smaller 20k map. However, the exact cause for this is the subject of ongoing research.

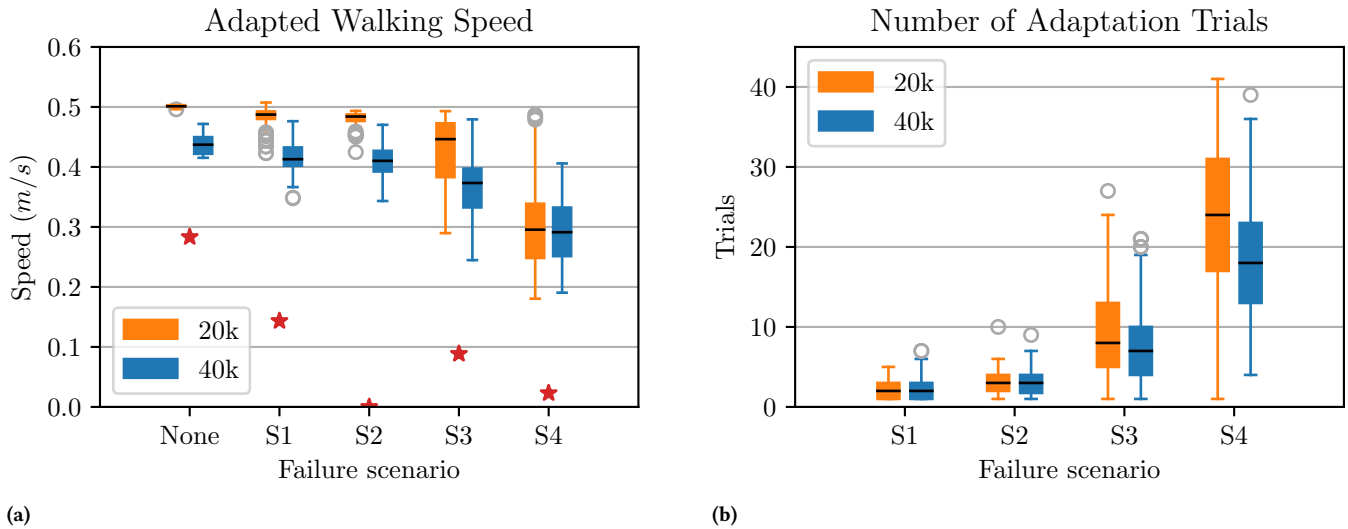


Figure 6: Simulated adaptation performance (a) Walking performance after adapting to failures in simulation. The red stars represent the mean gait performance of the reference tripod gait. (b) Number of trials required to adapt to failures in simulation. Box plots show distributions across 20 independently generated maps and all failure orientations.

6.2 Reality

Evolved controllers were successfully transferred to the physical robot and in the case of the undamaged scenario in figure 7a, enabled the hexapod to walk faster than the reference tripod gait. Gait performance was lower for all failure scenarios on the physical compared to the simulated robot. The performance drop also appears to be similar between failure scenarios. During the experiments foot slippage was noted and can be seen in the experiment videos (appendix A.3). This results from different floor surface properties in simulation and the laboratory where gaits evolved in simulation could rely on the relatively high coefficient of friction of 0.7 for grip. Also, 9–12 trials are required to find an optimal gait even without any failures. This is substantially more than the 2-5 trials required in [9], suggesting a slightly larger reality gap. However, even with the reality gap and different flooring material, the M-BOA adaptation algorithm resulted in the hexapod walking 2x faster than the reference tripod gait for both failure scenarios S1 and S2, shown in figure 7a. These results show the custom robot and gait controller are successfully able to failure scenarios S1 and S2 (section 5.1).

7 CONCLUSION

Results indicate a distinct trade-off between maximising adapted performance for simpler failures and minimising the number of trials (adaptation duration) for more difficult failures, thus elucidating the extent of adaptability given multiple leg failures for a hexapod robot, and highlighting the influence of the behaviour-performance map. We also presented new methods for integrating IT&E with an existing kinematic and trajectory based gait controller using fundamental components of statically stable legged locomotion which we validated on our physical hexapod robot. This controller allows for significantly refined and intuitive control over foot motions, while still exhibiting a level of behavioural diversity required to suitably adapt gaits to leg damage using IT&E.

In future work, we plan to investigate even smaller map sizes to determine where reduced map diversity begins to become detrimental to adaptation task performance. We also plan to conduct significantly more experiments on the physical robot to determine whether the same results are evident in reality.

ACKNOWLEDGMENTS

The authors acknowledge this research was partially funded by the South African *National Research Foundation* (NRF): Human and Social Dynamics in Development Grant (Grant No. 118557), with the aid of the *Center for High Performance Computing* (CHPC), South Africa, for providing computational resources to this project.

REFERENCES

- [1] P. Arm et al. 2019. SpaceBok: A Dynamic Legged Robot for Space Exploration. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, Montreal, Canada, 6288–6294. <https://doi.org/10.1109/ICRA.2019.8794136>
- [2] M. Bajracharya, M. Maimone, and D. Helmick. 2008. Autonomy for Mars Rovers: Past, Present, and Future. *Computer* 41, 12 (2008), 44–50.
- [3] J. Bares and D. Wettergreen. 1999. Dante II: Technical Description, Results, and Lessons Learned. *International Journal of Robotics Research* 18, 7 (1999), 621–649.
- [4] J Bellingham and K Rajan. 2007. Robotics in Remote and Hostile Environments. *Science* 318, 5853 (2007), 1098–1102.
- [5] J. Carlson and R. Murphy. 2005. How UGVs Physically Fail in the Field. *IEEE Transactions on Robotics* 21, 3 (2005), 423–437.
- [6] K. Chatzilygeroudis and J-B. Mouret. 2018. Using Parameterized Black-Box Priors to Scale Up Model-Based Policy Search for Robotics. In *ICRA 2018 - International Conference on Robotics and Automation*. IEEE, Brisbane, Australia, 5121–5128.
- [7] R. Christopher. 2019. *Mathematical Modelling and Control System Development of a Remote Controlled, IMU Stabilised Hexapod Robot*. Master’s thesis. University of Cape Town, South Africa.
- [8] E. Coumans and Y. Bai. 2016–2019. PyBullet, a Python module for physics simulation for games, robotics and machine learning. <http://pybullet.org>.
- [9] A. Cully, J. Clune, D. Tarapore, and J-B. Mouret. 2015. Robots that can Adapt like Animals. *Nature* 521, 7553 (2015), 503–507.
- [10] L. Dalcin, R. Paz, and M. Storti. 2005. MPI for Python. *J. Parallel and Distrib. Comput.* 65, 9 (2005), 1108–1115.
- [11] A. Davids. 2002. Urban Search and Rescue Robots: From Tragedy to Technology. *IEEE Intelligent Systems* 17, 2 (2002), 81–83.

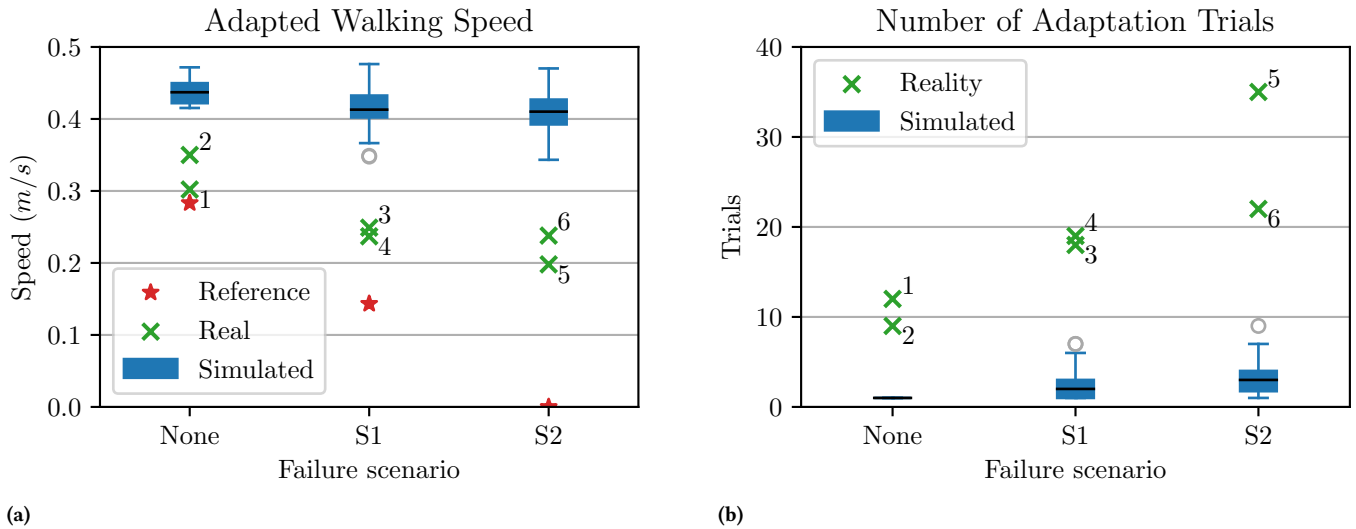


Figure 7: Real adaptation performance (a) Gait performance after adapting to failures in reality compared to simulation. Box plots repeat the simulated data from the 40k maps in figure 6a for comparison (b) Trials required to adapt to failures in reality compared to simulation. Green crosses are gait performance of real robot after adaptation. Red stars are mean gait performance of the reference tripod gait. The numbers correspond to the experiment numbers in table 2.

- [12] K. Deb and H-G. Beyer. 2001. Self-Adaptive Genetic Algorithms with Simulated Binary Crossover. *Evolutionary Computation* 9, 2 (2001), 197–221.
- [13] X. Ding, Z. Wang, A. Rovetta, and J. Zhu. 2010. Locomotion Analysis of a Hexapod Robot. In *Climbing and Walking Robots*. InTechOpen, London, UK, 291–310.
- [14] A. Eiben and J. Smith. 2015. *Introduction to Evolutionary Computing - Second Edition*. Springer, Berlin, Germany.
- [15] B. Flannery, S. Teukolsky, and W. Vetterling. 1986. *Numerical Recipes*. Cambridge University Press, Cambridge, UK.
- [16] P. Frazier. 2018. A Tutorial on Bayesian Optimization. arXiv:1807.02811 [stat.ML]
- [17] M. Hildebrand. 1989. The Quadrupedal Gaits of Vertebrates: The Timing of Leg Movements Relates to Balance, Body Shape, Agility, Speed, and Energy Expenditure. *BioScience* 39, 11 (1989), 766.
- [18] R. Isermann. 2006. *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Springer Science & Business Media, London, UK.
- [19] R. Kaushik, T. Anne, and J-B. Mouret. 2021. Fast Online Adaptation in Robotics through Meta-Learning Embeddings of Simulated Priors. arXiv:2003.04663 [cs.RO]
- [20] R. McGhee and A. Frank. 1968. On the Stability Properties of Quadruped Creeping Gaits. *Mathematical Biosciences* 3 (1968), 331–331.
- [21] J-B. Mouret. 2020. Python3 Map-Elites. github.com/resibots/pymap_elites.
- [22] J-B. Mouret and J. Clune. 2015. Illuminating Search Spaces by Mapping Elites. arXiv:1504.04909 [cs.AI]
- [23] A. Nagabandi, I. Clavera, S. Liu, R. Fearing, P. Abbeel, S. Levine, and C. Finn. 2019. Learning to Adapt in Dynamic, Real-World Environments Through Meta-Reinforcement Learning. arXiv:1803.11347 [cs.LG]
- [24] K. Nagatani and et al. 2013. Emergency Response to the Nuclear Accident at the Fukushima Daiichi Nuclear Power Plant using Mobile Rescue Robots. *Journal of Field Robotics* 30, 1 (2013), 44–63.
- [25] R. Putter and G. Nitschke. 2017. Evolving Morphological Robustness for Collective Robotics. In *IEEE Symposium Series on Computational Intelligence*. IEEE, Honolulu, USA, 1104–1111.
- [26] R. Putter and G. Nitschke. 2018. Objective versus Non-Objective Search in Evolving Morphologically Robust Robot Controllers. In *IEEE Symposium Series on Computational Intelligence*. IEEE, Bengaluru, India, 2033–2040.
- [27] ROBOTIS. 2006. *Dynamixel RX-28 User Manual*. Seoul, Korea.
- [28] X. Song, Y. Yang, K. Choromanski, K. Caluwaerts, W. Gao, C. Finn, and J. Tan. 2020. Rapidly Adaptable Legged Robots via Evolutionary Meta-Learning. arXiv:2003.01239 [cs.RO]
- [29] STMicroelectronics. 2020. *STM32F405xx STM32F407xx Datasheet*. Geneva, Switzerland.
- [30] D. Tarapore, J. Clune, A. Cully, and J-B. Mouret. 2016. How Do Different Encodings Influence the Performance of the MAP-Elites Algorithm?. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016* (Denver, Colorado, USA) (GECCO '16). Association for Computing Machinery, New York, NY, USA, 173–180. <https://doi.org/10.1145/2908812.2908875>
- [31] V. Vassiliades, K. Chatzilygeroudis, and J-B. Mouret. 2017. Using Centroidal Voronoi Tessellations to Scale Up the Multi-dimensional Archive of Phenotypic Elites Algorithm. arXiv:1610.05729 [cs.NE]
- [32] A. Winkler, C. Bellicoso, M. Hutter, and J. Buchli. 2018. Gait and Trajectory Optimization for Legged Systems Through Phase-Based End-Effector Parameterization. *IEEE Robotics and Automation Letters* 3, 3 (2018), 1560–1567.
- [33] Xsens. 2009. *MTi and MTx User Manual and Technical Documentation*. Enschede, Netherlands.

A APPENDICES

A.1 MAP-Elites Parameters

- Map dimensions: 6
- Controller dimensions: 32
- Number of niches: 20000, 40000
- Evaluations: 4×10^7
- Batch size: 2390
- Random initialisation: 1% of niches (Default)

A.2 M-BOA Parameters

- σ_{noise}^2 : 0.001
- α : 0.9
- ρ : 0.4
- κ : 0.05

A.3 Supplementary Materials

Experiment source code, data, and videos are available online at: https://github.com/chrismaier/maier_gecco_2021