Automating Damage Recovery in a Legged Robot

Alexandros Pouroullis, David Blore, Michael Scott
Julius Smith, Sindiso Mkhatshwa, Geoff Nitschke
Department of Computer Science,
University of Cape Town, South Africa
PRLALE004@myuct.ac.za, BLRDAV002@myuct.ac.za, SCTMIC015@myuct.ac.za
SMTJUL022@myuct.ac.za, MKHSIN035@myuct.ac.za, gnitschke@cs.uct.ac.za

***Abstract*—Autonomous robots are increasingly used in remote and hazardous environments, where automated recovery given damage to sensory-actuator systems would be extremely beneficial. Such robots must therefore have controllers that continue to function effectively given unexpected hardware malfunctions and damage. We evaluate various controller types (oscillator-style central pattern generators and artificial neural networks), for producing adaptable gait behaviors. These controller types are run for hexapod robot gait control in concert with the *Intelligent Trial and Error* (IT&E) and Map-Elites algorithm to maintain behavioral diversity. Specifically, we investigate the impact of behavior map-size in MAP-Elites (the first phase of the IT&E algorithm), in company with various controller types for multiple leg failures scenarios using a simulated hexapod robot. Results support previous work demonstrating a trade-off between adapted gait speed and controller adaptability across leg-damage scenarios, where map-size is crucial for generating behavioral diversity required for adaptation.**

***Index Terms*—Evolutionary Robotics, Behavioral Diversity**

I. Introduction

As autonomous robots are deployed for various complex tasks in remote, unpredictable, and hazardous environments [1], the risk of incurring hardware damage and failing to complete tasks increases [2]. Thus, a pertinent problem is how to generate effective controllers that adapt to hardware damage [3], [4], such that the robot continues to function [5]. This is especially pertinent in environments where *in situ* repair is not possible [6]. Legged robotic gait evolution is an established benchmark in evolutionary robotics, where task performance is typically evaluated as locomotion speed and number of generations (evaluations) required to evolve effective gaits [7]. Previous work investigated gait evolution across various robotic platforms (bipeds [8], quadrupeds [9], and hexapods [10]), but largely ignored gait adaptation given changing morphologies (due to incurred sensor-actuator damage during the robot's runtime [5]). Evolutionary controller adaptation is effective for robot behavior adaptation given sensor-actuator (morphological) damage [5], [7], [11], [12]. For example, *Intelligent Trial and Error* (IT&E) and Map-Elites [5] have adapted hexapod locomotion under various leg failure scenarios [5], [10]–[13]. IT&E and Map-Elites have adaptability benefits (suitably balancing exploration of new gaits versus exploitation of functional gaits), over comparative robotic gait adaptation methods including model-based reinforcement learning [14], [15]. Another benefit of IT&E and Map-Elites is that it generates adapted behaviors for various controller types, such as oscillator-style *Central Pattern Generators* (CPGs) [16] and evolved neural networks [17]. However, the impact of various controller encodings in company with specific IT&E and Map-Elites parameters on behavioral adaptation given morphological damage, remains relatively unexplored [12].

Previous gait adaptation work using IT&E and Map-Elite indicated behavior map-size impacts adaptive gait performance [12], where small versus large map-sizes shifted behavioral (gait) adaptation from exploitation to exploration. Though, previous work investigated few map-sizes with one controller type. Indirect controller encodings such as HyperNEAT [18] have also been demonstrated for gait adaptation over various robot morphologies [19], [20]. Thus, this study investigate the adaptability of various controller encodings with behavioral diversity maintenance (Map-Elites behavior map-sizes), given various leg-damage scenarios. Given demonstrated effectiveness of Map-Elites in related work [5], [10], [12], we focus on Map-Elites controller evolution for adapting hexapod locomotion to leg-damage.

We evaluate five hexapod robot controller encodings for adaptive gait behavior. All use Map-Elites [5] for behavioral diversity maintenance and were evaluated for various behavior-map sizes and damage scenarios. First, we evaluate a kinematic-trajectory based open-loop controller [12]. Second, a CPPN [21] encoded open-loop controller using non-linear oscillators (CPGs) [22]. Open-loop controllers were selected for these controller types given already demonstrated efficacy for generating highly adaptable gaits [5], [7], [11], [23]. Third, we evaluated CPPN connection weight encoding of an artificial neural network (ANN) controller [24]. Fourth, we evaluated an ANN controller, where hidden-layer topology and connection weights were evolved by NEAT [25]. NEAT, a direct-encoding neuro-evolution method, was selected to gain insights into potential benefits of indirect versus direct controller encodings, when used with various Map-Elites behavior-map-sizes. Finally, we evaluated a CPPN encoded *Single Unit Pattern Generator* (SUPG) [7].

We also address the limitations of related work on hexapod gait adaptation [12], offering three contributions. First, we comparatively evaluate five controller types for a range of behavior-map sizes and leg damage scenarios, whereas related work only evaluated an open-loop kinematic controller. Second, we evaluate Map-Elites behavior-map sizes: [5k, 10k, 20k, 40k] per controller and damage scenario, whereas related work only tested two map sizes for one controller. Third, we present insight into which behavior-map size, controller type coupling elicits the most adaptable gaits (Section IV).

II. Methods

The core methods were five hexapod controller encodings (Section II-A$-$II-E), adapted with IT&E and Map-Elites (Section II-F). Evolved gaits were then optimized *online* during hexapod runtime (Section II-G). Iterated Racing for Automatic Algorithm Configuration [26] automated parameter tuning for all controllers, Map-Elites and online optimization.

A. Reference (Open-Loop) Controller

The reference (kinematic) gait controller and all controller parameters are as in related work [12]. 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 sixth order polynomial for the swing phase foot motion. Foot trajectories are transformed into joint angles using inverse-kinematics and to servos at 120 Hz. This controller used an open-loop configuration with base stabilisation turned off. Controller parameterization and leg trajectories used related work [27], including fundamental components of statically stable legged gaits such as support polygon [28] and foot timing [29], thus producing diverse symmetrical and asymmetrical hexapedal gaits [30].

B. CPG (Open-Loop) Controller

The CPG controller extends previous work [7], where outputs were scaled to match the hexapod's joint limits [12], and comprised of 18 coupled, amplitude-controlled phase oscillators determining the joint angle of three servos per leg on the hexapod. Oscillator behavior equations and controller parameters can be found in Tarapore et al. [7].

C. CPPN Encoded CPGs

A CPPN encodes the intrinsic amplitudes A_i and inter-oscillator phase biases $\phi_{i,j}$ of 18 CPG oscillators. Oscillators are configured as a 2D Cartesian grid (substrate), so each oscillator has a distinct (x,y) coordinate reflecting hexapod morphology [7]. The intrinsic amplitude per oscillator i is used for CPPN inputs: (x_i,y_i). Amplitude outputs are scaled to the allowable angular range of corresponding motors [12]. CPG oscillators are coupled, so as phase bias for every pair of adjacent oscillators (i, j) is a CPPN query with inputs (x_i, y_i) and (x_j, y_j), and scaled output range: [0, 2π]. CPG constraints are as in previous work [7], with 18 intrinsic amplitude and 18 phase bias parameters encoded by the CPPN.

D. ANN Controller Encodings

ANNs used 20 input, 18 output and hidden nodes where node connectivity and weights were *directly* encoded and NEAT evolved [25] or *indirectly* (CPPN [21]) encoded and HyperNEAT evolved [18]. ANN inputs were 18 angles per leg servo (s_1, s_2, s_3) and *sine*, *cosine* wave functions at 1 Hz frequency, per simulator time step and multiplied by π for periodic gait behaviors. ANN output, per time-step, were 18 values ([-1, 1]), for each servo (s_1, s_2, s_3), scaled to allowable angular ranges [12]. Outputs were hexapod servo motor angles per time step. Hidden node inputs were initially 0 so the input layer was (initially) directly connected to the output layer. For direct encoding, ANN connection weights and hidden layer connectivity was evolved by NEAT [25]. For indirect encoding, NEAT evolved CPPNs [18], [31], encoding fixed topology recurrent ANNs (outputs connecting back to inputs). These ANNs used 20 inputs (s_1, s_2, s_3 per leg, and *sine*, *cosine* inputs for periodic gait behaviors [7]), 18 hidden and 18 output nodes, where ANN (substrate) node positions corresponded to hexapod morphology [7]. The CPPN was iteratively queried with the positions of all source (x_1, y_1) and target (x_2, y_2) nodes, where CPPN output corresponded to weights of input-hidden and hidden-output neuron connections. ANN weights were a function of geometric coordinates of the source and target node per connection. ANN node coordinates and a constant bias were iteratively passed to the CPPN to determine each connection weight. Given no hidden layer, the CPPN had only one output (the weight between the source input layer node and output layer target node). Given a hidden layer, the CPPN had two output values, specifying weights for each connection layer [18]. At each time-step, the ANN was input with previous time-step angle values for the servos (s_1, s_2, s_2) on each leg and a *sine* and *cosine* wave. ANN output nodes specified new joint angles per leg servo (s_1, s_2, s_2), per time-step. For both encodings, all connection weights were randomly initialized as in previous work [11].

E. Encoding SUPGs with CPPNs

Our SUPG produces repeated cycles of CPPN encoded activation patterns generating temporal oscillations [11]. Three (coupled) SUPGs were used per hexapod leg (18 total), applied to *coxa* and *femur* joints for joint angle output per time-step. CPPN input is the position (x, y) of the SUPG in the substrate and elapsed time ([0.0, 1.0]) since the SUPG was last triggered. Elapsed time, a SUPG internal timer, increases from 0 to a maximum of 1 (per period of SUPG output). SUPG output is a function of its substrate position and the time since its last cycle was triggered, meaning SUPG outputs specify the desired angles per servo (s_1, s_2), per leg. The *tibia* joint output is a function of the *femur* output such that tibia tip is always pointing to the ground. SUPG output per time-step is then used as CPPN input at the next time-step. For time-step 1, neutral angle positions are used as CPPN input. The SUPG's internal timer is restarted from 0 given an external trigger event, where the oscillation period is adjusted to match hexapod gait and terrain by restarting the SUPG whenever its associated foot touches the ground, producing closed-loop control [7]. The three SUPGs actuating each leg were simultaneously triggered by the corresponding foot touching the ground. At time-step 1, all legs touched the ground, so all SUPGs were triggered simultaneously. To avoid *hopping* gaits, the first trigger to each SUPG was delayed by an offset. The CPPN offset output was determined for the s_1 SUPG per leg by supplying its coordinates as input. The same offset value was applied to s_2 SUPG per leg, so as oscillators per leg start concurrently. Since the SUPG is capable of oscillatory signal frequencies exceeding 1Hz [7], and high frequency gaits over-tax robot motor-servos [32], we constrained SUPG oscillatory signal frequency to 1Hz, and the CPPN with summed weights between timer-input and phase-output nodes to a maximum of 2π radians. SUPG controller gaits were then evolved with the constrained-CPPN using NEAT (Section II-D).

F. Behavior (Gait) Map Generation

IT&E's first stage is generating diverse gaits (quality-diversity map), representing behaviors used for adaptation [5]. The map is generated using *Centroidal Voronoi Tessellation* (CVT)-MAP-Elites [33], [34]. CVT-MAP-Elites begins by discretizing a behavior space into k evenly spaced niches with a CVT for uniform distribution of behaviors. Once k niches were created, the map $(\mathcal{X}, \mathcal{P})$ is initialized to store solutions $(\mathcal{X})$ and task performances $(\mathcal{P})$. A random population (G) of solutions $(\mathbf{x})$ was initialized in the map, where a solution represented a given controller behavior. Evaluated controllers elicit a task performance (p, Section III-A) and behavior descriptor $(\mathbf{b})$ for evolved gaits. The behavior descriptor (as in related work [12]) determines the behavior per map-niche. After random initialization MAP-Elites follows a parent selection and genetic variation loop [35], with random solution selection and variation using *Simulated Binary Crossover* (SBX) [36]. Varied solutions $(\mathbf{x}')$ are added to the map if the niche is empty $(\mathcal{P}(c) = \emptyset)$ or the current niche (solution) is lower performance $(\mathcal{P}(c) < p)$. Selection and variation ran for 40 million evaluations (as per related work parameter settings [5]) to fill the map with gait behaviors. Other Map-Elites parameter settings were as in previous work [12]. The second stage of IT&E is gait adaptation (Section II-G), occurring *online* during hexapod operation (gait adaptation task, Table II).

G. Gait Adaptation

Given leg damage, the hexapod adapts via searching the behavioral map (Section II-F) for a new gait using the *Map-based Bayesian Optimisation Algorithm* (M-BOA) [5]. Gait performance prediction (P) is modeled with a Gaussian process $(\mathcal{N})$, using the expected performances from the map $(\mathcal{P})$ as a prior [37]. The gait predicted to perform the best is selected using the *upper confidence bound* (UCB) function. Predicted gaits are trialed on the robot and gait performance (p_{t+1}) updates the prediction model. This process repeats, improving gait performance prediction, terminating if actual performance (p_{t+1}) is within α of maximum predicted performance. Evaluation metrics were gait speed and trials required

TABLE I: M-BOA and MAP-Elites Parameters.

M-BOA	σ^2_{noise}:	0.001
	α:	0.9
	ρ:	0.4
	κ:	0.05
MAP-Elites	Map dimensions:	6
	Controller dimensions:	32
	Number of niches:	5000, 10000, 20000, 40000
	Evaluations:	4×10^7
	Batch size:	2390
	Random initialization:	1% of niches (Default)

for adaptation, with each gait trial lasting 5 seconds (Table I). M-BOA gait performance feedback (p_{t+1}) used related work parameters [5], and the task performance metric (Section III-A) and simulator used for map generation (Section II-F).

III. Experiments

Experiments simulate a custom hexapod robot[1], with 18 degrees-of-freedom, where each leg has joints (servos) at the *coxa*, *tibia* and *femur* joints [12]. Our simulator uses the *PyBullet* [38] physics engine running at 240 Hz, using a *Unified Robotics Description Format* (URDF) hexapod model. Experiments evaluated the impact of five controller types and Map-Elite map-sizes (5k, 10k, 20k and 40k niches), given IT&E and Map-Elites adapted gait performance over four leg failure scenarios (S1−S4, Table II), and a benchmark no leg-damage scenario (S0, Table II). Each controller type was evaluated for 5 seconds (1200 simulation time-steps), over the leg failure scenarios, where leg failure was approximated by locking a leg in a retracted position. Per controller type and map-size coupling, average gait performance (Section III-A) was calculated over 20 runs for each failure scenario.

A. Task Performance Function

Gait performance (p) was the average velocity of the center of the hexapod along a single axis. Velocity along a single axis encouraged gaits exhibiting straight line motion in a forward direction. To discourage unsafe gaits the following conditions resulted in simulation termination and a task performance of 0 m/s: Collisions between the legs, collisions between the base and the ground, and leg kinematic singularities.

B. Behavior-Performance Map Size

Behavior-performance map size in Map-Elites is determined by the number of niches (k), controlling overall behavioral (gait) map diversity. Exploratory experiments indicated map-sizes over 40k niches, yielded negligible task performance increases, thus we tested relatively small map-sizes (5k−40k) in comparison to related work [5], [11], [12], [33].

[1]Experiment source code, data, and videos are available online: https://github.com/AlePouroullis/SSCI2025_Hexapod

TABLE II: Failure Scenario and Controller Adaptation Experiments

Experiment	Failure Scenario	Controller Type	Map-Sizes
1		SUPG	[5k, 10k, 20k, 40k]
	S1:	Reference [12]	[5k, 10k, 20k, 40k]
	One failed leg	CPG	[5k, 10k, 20k, 40k]
		ANN (NEAT)	[5k, 10k, 20k, 40k]
		ANN (HyperNEAT)	[5k, 10k, 20k, 40k]
2		SUPG	[5k, 10k, 20k, 40k]
	S2:	Reference [12]	[5k, 10k, 20k, 40k]
	Two failed legs	CPG	[5k, 10k, 20k, 40k]
	seperated	ANN (NEAT)	[5k, 10k, 20k, 40k]
	by two functional legs	ANN (HyperNEAT)	[5k, 10k, 20k, 40k]
3		SUPG	[5k, 10k, 20k, 40k]
	S3:	Reference [12]	[5k, 10k, 20k, 40k]
	Two failed legs	CPG	[5k, 10k, 20k, 40k]
	seperated	ANN (NEAT)	[5k, 10k, 20k, 40k]
	by one functional leg	ANN (HyperNEAT)	[5k, 10k, 20k, 40k]
4		SUPG	[5k, 10k, 20k, 40k]
	S4:	Reference [12]	[5k, 10k, 20k, 40k]
	Two adjacent	CPG	[5k, 10k, 20k, 40k]
	failed legs	ANN (NEAT)	[5k, 10k, 20k, 40k]
		ANN (HyperNEAT)	[5k, 10k, 20k, 40k]

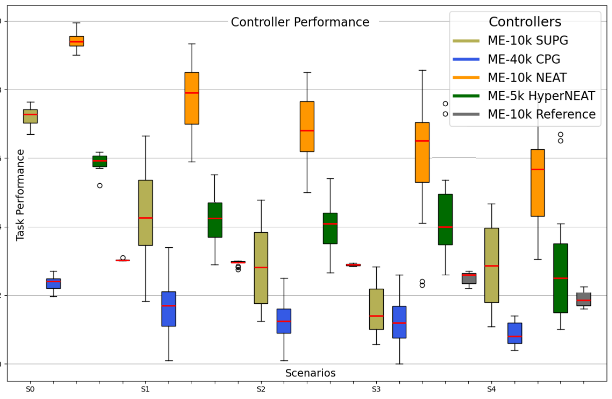

Fig. 1: Average maximum (20 runs) gait quality (normalized fitness: [0.0, 1.0]) per fittest controller for no-damage (S0) and damage (S1−S4) scenarios (legend: controller type, map-size).

C. *Gait Behavior Quality Metrics*

To gauge evolved gait quality-diversity per method (controller, map-size coupling), we measured maximum *behavior quality* (gait speed), and computed the *Quality-Diversity* (QD) score [39] per map for the best performing gait (for a given damage scenario). The QD score was calculated as the overall *quality* (task performance) in filled cells within a map corresponding to the highest performing behavior (gait) for a given run, and the average QD score computed over 20 runs. A high average QD score indicates controllers yield a high average behavioral diversity (gait types) and quality (gait speed).

IV. Results and Discussion

MAP-Elites was run for 20 runs (40 million evaluations per run) per controller type and map-size (5k, 10k, 20k and 40k), and average maximum gait speed (Section III-A) computed

TABLE III: Average (20 runs, all damage scenarios) QD scores per controller and map-size. Overall highest scores in bold.

	Reference	CPG	SUPG	NEAT	HyperNEAT
QD Score	0.52	0.15	0.70	0.89	**0.75**
Map-size	5k	5k	5k	5k	**5k**
QD Score	**0.61**	0.10	**0.72**	**0.90**	0.65
Map-size	**10k**	10k	**10k**	**10k**	10k
QD Score	0.55	0.11	0.61	0.78	0.68
Map-size	20k	20k	20k	20k	20k
QD Score	0.57	**0.19**	0.65	0.74	0.69
Map-size	40k	**40k**	40k	40k	40k

per leg failure scenario. For all results, Kolmogorov–Smirnov normality tests with Lilliefors correction [40] indicated non-parametric data. Mann–Whitney U statistical tests [41] were then applied in pair-wise comparisons with Effect Size [42] treatment. Figure 1 presents box-plots of average maximum task performance (*gait speed*) per controller per damage scenario (S0−S4, Table II). Figure 1 illustrates that for all leg-damage scenarios (Table II), the NEAT controller (map-size: 10k) yielded highest average gait quality (0.68 of maximum gait speed, Figure 1). Figure 2 presents QD maps (computed over 20 runs) for the highest *quality* (fastest) gaits across damage scenarios (S1−S4, Table II) evolved by IT&E and Map-Elites per controller. Gait *diversity* is indicated by filled niches (shaded regions) and color intensity represents gait quality. Map-sizes yielding gaits with the highest quality-diversity are: 10k, 40k, 10k, 5k and 10k for the Reference, CPG, NEAT, HyperNEAT and SUPG controllers, respectively.

Figures 1 (average gait quality), 2 (average quality-diversity) indicate, the highest quality gaits across leg-damage scenarios, were yielded for *small* (10k) map-sizes. Each

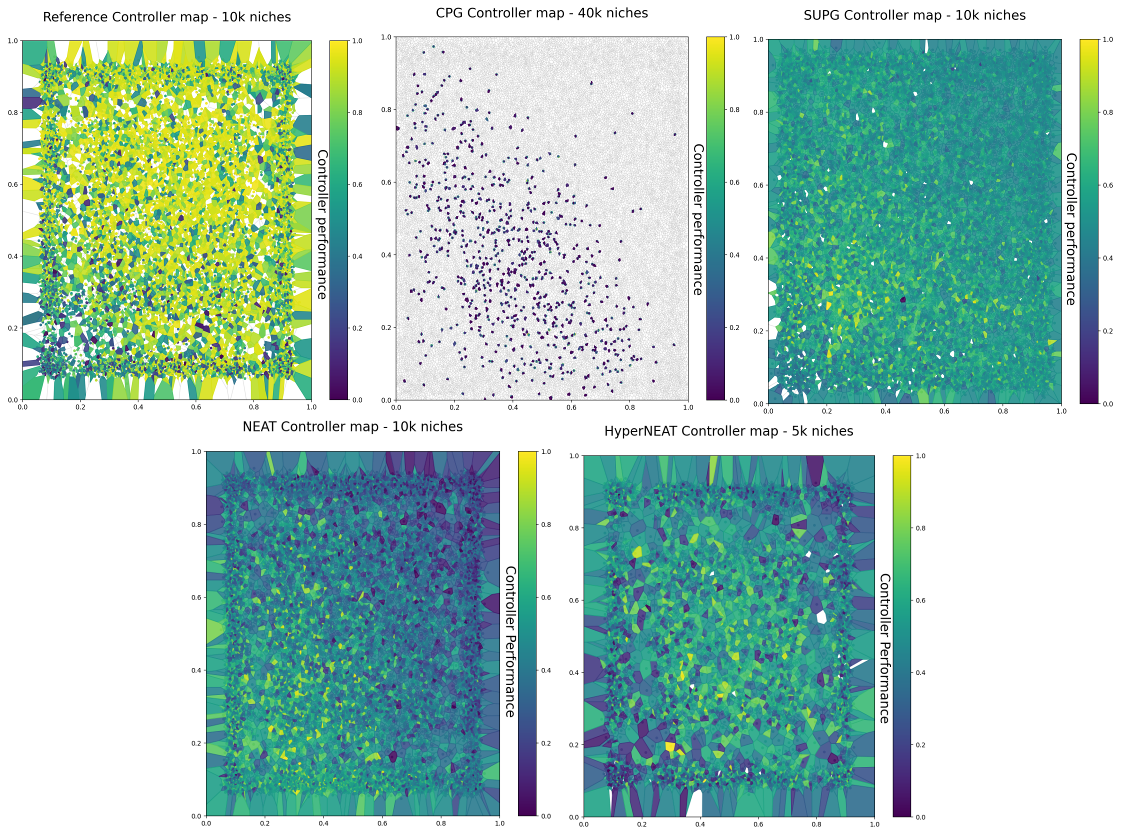

Fig. 2: Average (20 runs, all leg-damage scenarios) QD maps for fittest gaits using: (top-left−bottom-right) Reference, CPG, SUPG, NEAT, HyperNEAT controllers (map-size for each indicated). Gait *diversity* is indicated by niches filled. Gait *quality* is indicated by color intensity (lighter shade: higher quality gaits, darker shade: lower quality gaits).

controller type evolved using 5K, 10K map-size yielded significantly higher ($p < 0.05$) average gait quality compared to controllers evolved using larger (20k, 40k) map-sizes across leg-damage scenarios. The exception was the CPG evolved with a 40k map-size. However, the CPG yielded a significantly lower ($p < 0.05$) average gait quality compared to other controllers (Figure 1). This is consistent with previous results [12], [33], attributing increased selection pressure and higher gait quality given fewer search space niches and thus lower gait quality given more search space niches.

Average QD maps per controller type and leg-damage scenarios (Figure 2) further support the benefits of MAP-Elites controller evolution using small map-sizes (Figure 1). The overall highest average QD score (0.90 for NEAT, map-size: 10k, Table III) was significantly higher ($p < 0.05$) than average QD scores for comparative controllers for map-sizes: 5k, 20k, 40k (Table III). NEAT evolved controllers using these larger map-sizes still yielded significantly higher ($p < 0.05$) average QD scores compared to other controllers evolved with the same map-sizes (Table III). For the given leg-damage scenarios, this supports the suitability of NEAT evolved controllers and MAP-Elites behavior diversity maintenance using small map-sizes. Supporting the benefit of small maps-sizes in Map-Elites behavioral diversity maintenance, the Reference, SUPG and HyperNEAT controllers similarly yielded their highest average gait quality when coupled with 10K map-sizes (5K for HyperNEAT). Furthermore, QD maps (Figure 2) indicate that if the map-size is too small then this inhibits the discovery of consistently high-quality and adaptable gaits across leg-damage scenarios. The exception was HyperNEAT (highest QD score overall for a 5k map-size, Table III). All other controllers evolved with 5k map-size yielded significantly lower ($p < 0.05$) average QD scores. Results thus indicate specific map-sizes mitigate the behavior space exploitation versus exploration trade-off, highlighting a relationship between direct controller encoding, map size and controller adaptability across damage scenarios.

Direct controller encoding benefits are also evidenced by comparing QD search efficacy of NEAT versus HyperNEAT evolved gaits, across leg-damage scenarios. Given the QD maps generated for the highest quality NEAT and HyperNEAT evolved gaits (Figure 2), one observes visible gaps (white-space) in the search-space coverage of HyperNEAT despite using a smaller map-size (5k) compared to NEAT (10k) for the same damage scenarios. Comparing the search space coverage of the highest quality Reference, CPG and SUPG controllers, we observe lower quality-diversity (fewer niches filled and darker shading per niche, Figure 2) for the QD maps corresponding to gaits evolved by these controllers. This observed lower quality-diversity of the Reference, CPG and SUPG controllers is also supported by the significantly lower QD scores ($p<0.05$) of these controller gaits versus the NEAT gait (Table III). Thus NEAT's direct encoding evolution coupled with Map-Elites behavioral diversity maintenance is suitable for discovery of sensory-actuator patterns (irregular but coordinated leg movements) with high quality gaits for all damage scenarios. Behavioral diversity maintenance benefits are also indicated by NEAT's full coverage of the QD-map (Figure 2) and the highest average QD score (Table III).

This contrasts to related work [7], [11], [20] indicating HyperNEAT's indirect controller evolution (with Map-Elites behavioral diversity maintenance) affords discovery of highly adaptable gaits. However, this considered HyperNEAT's indirect encoding evolution beneficial for adapting gait *regularity* in legged robots [24], [43], [44] given *regular* tasks, where robots had no (leg) damage. Whereas, our results extend work on direct versus indirect controller evolution for legged robot adaptation [45] across *regular* (no leg-damage) and *irregular* (leg-damage) scenarios. We define *regularity* versus *irregularity* in-line with work on regularity of structure compressibility (reducible repeating components) [46] and symmetry [47]. Specifically, we define gait adaptation as *regular*, given hexapod *symmetry*: equal numbers of legs along the hexapod's horizontal axis (spine). Similarly, the task is *irregular* given an unequal number of legs along the hexapod's spine. Thus, we define irregularity in these gait adaptation tasks as the number of disabled legs, where irregularity increases with leg-damage (Table II), and the hexapod had to adapt its gait to asymmetrical movement.

Whereas related work leveraged HyperNEAT's capability to exploit task-based geometric properties [18], [24], [43], [44] such as symmetry and regularity in adapted locomotion, in this case HyperNEAT's core benefits were not suitable for evolving behavioral irregularities necessary to generate stable gaits on irregular damage scenarios. Similarly, results indicate that parameterized controllers (Reference, CPG and SUPG: Sections II-A−II-E), evolved with Map-Elites behavioral diversity maintenance, were ineffective for producing gaits adaptable across leg-damage scenarios (Figures 1, 2). Similar to HyperNEAT's evolutionary design method, this is theorized to result from such parameterized controllers being specifically designed produce rhythmic body movements using artificial neuron based central pattern generators that act as nonlinear oscillators to produce stable gaits via coordinated motor-neuron oscillations [48]. These parameterized controllers exploit robot morphological symmetry to produce cyclic gait (leg-actuator) patterns to generate a stable and fast gaits [12]. However, such benefits are diminished due to the irregular leg-damage scenarios (Table II), and the irregular gaits required for high task performance (Section III-A).

This study presents two key contributions. First, the impact of the map-size parameter in behavioral diversity maintenance (Section II-F), coupled with controller evolution (gait adaptation) across leg-damage scenarios. Small map-sizes (10k), set as a Map-Elites behavioral diversity maintenance parameter, were beneficial for evolving gaits across leg-damage scenarios (Table II). Results also demonstrated that too small or too large of a map-size negatively impacts the task performance of adapted gaits generated across leg-damage scenarios. Second, NEAT's direct controller encoding was suitable for evolving asymmetrical gaits on irregular leg-damage scenarios (Table II), and thus the potential suitability of direct-encoding evolution and behavioral diversity maintenance for generating gaits that are suitable adaptations to robotic (morphological) damage. The adaptability of NEAT evolved controllers (with all map-sizes), was supported by the significantly higher task performance of adapted gaits across leg-damage scenarios. A key overall contribution is thus that evolving directly encoded ANN controllers in company with behavioral diversity maintenance using specifically (small) map-sizes mitigates the exploitation versus exploration trade-off in the evolutionary search for suitably adapted *irregular* gaits.

V. Conclusion

We evaluated the evolution of *Single Unit Pattern Generator* (SUPG), *Central Pattern Generator* (CPG) and direct (NEAT) and indirect (HyperNEAT) controller encodings, using *Intelligent Trial and Error* (IT&E) and *Map-Elites* to maintain behavioral diversity in evolved hexapod gaits. We evaluated the efficacy of these controller encodings on various *irregular* gait adaptation tasks (leg-damage scenarios). Results indicate that overall small Map-Elites maps-sizes coupled with a direct ANN controller encoding (NEAT) elicits maximal average task performance across a range of leg-damage scenarios. The core contribution of these results is two-fold. First, that use of behavioral map-sizes that are too small or too large of a map-size is disadvantageous to the evolutionary search for suitable gaits (adapted to leg-damage). Second, NEAT's direct controller encoding was more suitable, compared to comparative indirectly coded and parameterized controllers, for evolving asymmetrical gaits in response to leg-damage. Overall results indicate potential for the coupling of direct-encoding evolution and behavioral diversity maintenance given robotic controller damage recovery tasks. Ongoing work is investigating the impact of lifetime learning coupled with artificial evolution [49] on gait adaptation across complex task environments [50] for automating robot damage recovery with the objective of achieving self-sustaining robotic systems [51].

References

[1] J. Bellingham and K. Rajan, "Robotics in Remote and Hostile Environments," *Science*, vol. 318, no. 5853, pp. 1098–1102, 2007.

[2] J. Carlson and R. Murphy, "How UGVs Physically Fail in the Field," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 423–437, 2005.

[3] R. Putter and G. Nitschke, "Evolving Morphological Robustness for Collective Robotics," in *IEEE Symposium Series on Computational Intelligence*. Honolulu, USA: IEEE, 2017, pp. 1104–1111.

[4] ——, "Objective versus Non-Objective Search in Evolving Morphologically Robust Robot Controllers," in *IEEE Symposium Series on Computational Intelligence*. Bengaluru, India: IEEE, 2018, pp. 2033–2040.

[5] A. Cully *et al.*, "Robots that can Adapt like Animals," *Nature*, vol. 521, no. 7553, pp. 503–507, 2015.

[6] R. Isermann, *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. London, UK: Springer Science & Business Media, 2006.

[7] D. Tarapore and J.-B. Mouret, "Evolvability Signatures of Generative Encodings: Beyond Standard Performance Benchmarks," *Information Sciences*, vol. 313, no. 1, pp. 43–61, 2015.

[8] H. Liu and H. Iba, "A Hierarchical Approach for Adaptive Humanoid Robot Control," in *Proceedings of the Congress on Evolutionary Computation*. Portland, USA: IEEE, 2004, pp. 1546–1553.

[9] S. Risi and K. Stanley, "Confronting the Challenge of Learning a Flexible Neural Controller for a Diversity of Morphologies," in *Proceedings of the Genetic and Evolutionary Computation Conference*. Dublin, Ireland: ACM, 2013, pp. 255–262.

[10] M. Allard *et al.*, "Online damage recovery for physical robots with hierarchical quality-diversity," *ACM Trans. Evol. Learn. Optim.*, vol. 3, no. 2, jun 2023. [Online]. Available: https://doi.org/10.1145/3596912

[11] D. Tarapore *et al.*, "How Do Different Encodings Influence the Performance of the MAP-Elites Algorithm?" in *Proceedings of the Genetic and Evolutionary Computation Conference*. Denver, USA: ACM, 2016, pp. 173–180.

[12] C. Mailer, G. Nitschke, and L. Raw, "Evolving Gaits for Damage Control in a Hexapod Robot," in *Proceedings of the Genetic and Evolutionary Computation Conference*. Lille, France: ACM, 2021, pp. 146–153.

[13] K. Chatzilygeroudis, V. Vassiliades, and J.-B. Mouret, "Reset-free trial-and-error learning for robot damage recovery," *Robotics and Autonomous Systems*, vol. 100, no. 1, pp. 236–250, 2018.

[14] K. Chatzilygeroudis and J.-B. Mouret, "Using Parameterized Black-Box Priors to Scale Up Model-Based Policy Search for Robotics," in *Proceedings of the International Conference on Robotics and Automation*. Brisbane, Australia: IEEE, 2018, pp. 5121–5128.

[15] R. Kaushik, T. Anne, and J.-B. Mouret, "Fast Online Adaptation in Robotics through Meta-Learning Embeddings of Simulated Priors," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Las Vegas, USA: IEEE, 2020, pp. 5269–5276.

[16] J. Yu *et al.*, "A Survey on CPG-inspired Control Models and System Implementation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 3, pp. 441–456, 2014.

[17] S. Doncieux *et al.*, "Evolutionary Robotics: What, Why, and Where To," *Frontiers in Evolutionary Robotics*, vol. 2, no. 4, pp. 1–18, 2015.

[18] K. Stanley, D. D'Ambrosio, and J. Gauci, "A Hypercube-Based Encoding for Evolving Large-Scale Neural Networks," *Artificial Life*, vol. 15, no. 2, pp. 185–212, 2009.

[19] G. Lan *et al.*, "Learning Directed Locomotion in Modular Robots with Evolvable Morphologies," *Applied Soft Computing*, vol. 111, no. 1, p. 107688, 2021.

[20] K. Miras, "Constrained by Design: Influence of Genetic Encodings on Evolved Traits of Robots," *Frontiers in Robotics and AI*, vol. 8, no. 1, p. 672379, 2021.

[21] K. Stanley, "Compositional Pattern Producing Networks: A Novel Abstraction of Development," *Genetic Programming and Evolvable Machines*, vol. 8, no. 2, pp. 131–162, 2007.

[22] A. Ijspeert *et al.*, "From Swimming to Walking with a Salamander Robot Driven by a Spinal Cord Model," *Science*, vol. 315, no. 5817, pp. 1416–1420, 2007.

[23] A. Crespi *et al.*, "Salamandra Robotica II: An Amphibious Robot to Study Salamander-like Swimming and Walking Gaits," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 308–320, 2013.

[24] J. Clune *et al.*, "On the Performance of Indirect Encoding across the Continuum of Regularity," *IEEE Transactions on on Evolutionary Computation*, vol. 15, no. 4, pp. 346–367, 2011.

[25] K. Stanley and R. Miikkulainen, "Evolving Neural Networks through Augmenting Topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.

[26] M. López-Ibáñez *et al.*, "The irace Package: Iterated Racing for Automatic Algorithm Configuration," *Operations Research Perspectives*, pp. 43–58, 2016.

[27] A. Winkler *et al.*, "Gait and Trajectory Optimization for Legged Systems Through Phase-Based End-Effector Parameterization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, 2018.

[28] R. McGhee and A. Frank, "On the Stability Properties of Quadruped Creeping Gaits," *Mathematical Biosciences*, vol. 3, pp. 331–331, 1968.

[29] M. Hildebrand, "The Quadrupedal Gaits of Vertebrates: The Timing of Leg Movements Relates to Balance, Body Shape, Agility, Speed, and Energy Expenditure," *BioScience*, vol. 39, no. 11, p. 766, 1989.

[30] X. Ding *et al.*, "Locomotion Analysis of a Hexapod Robot," in *Climbing and Walking Robots*. London, UK: InTechOpen, 2010, pp. 291–310.

[31] N. Cheney *et al.*, "Unshackling Evolution: Evolving Soft Robots with Multiple Materials and a Powerful Generative Encoding," in *Proceedings of the Genetic and Evolutionary Computation Conference*. Amsterdam, Netherlands: ACM, 2013, pp. 167–174.

[32] J. Yosinski *et al.*, "Evolving Robot Gaits in Hardware: The HyperNEAT Generative Encoding vs. Parameter Optimization," in *Proceedings of the European Conference on Artificial Life*. Paris, France: ACM, 2011, pp. 890–897.

[33] V. Vassiliades, K. Chatzilygeroudis, and J.-B. Mouret, "Using Centroidal Voronoi Tessellations to Scale Up the Multi-Dimensional Archive of Phenotypic Elites Algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 4, pp. 623–630, 2018.

[34] J.-B. Mouret, "Python3 map-elites," github.com/resibots/pymap_elites, 2020.

[35] A. Eiben and J. Smith, *Introduction to Evolutionary Computing - Second Edition*. Berlin, Germany: Springer, 2015.

[36] K. Deb and H.-G. Beyer, "Self-Adaptive Genetic Algorithms with Simulated Binary Crossover," *Evolutionary Computation*, vol. 9, no. 2, pp. 197–221, 2001.

[37] P. Frazier, "A Tutorial on Bayesian Optimization," 2018.

[38] E. Coumans and Y. Bai, "PyBullet: A Python Module for Physics Simulation for Games, Robotics and Machine Learning," http://pybullet.org, 2019.

[39] J. Pugh *et al.*, "Confronting the Challenge of Quality Diversity," in *Proceedings of the Genetic and Evolutionary Computation Conference*. Madrid, Spain: ACM, 2015, pp. 967–974.

[40] A. Ghasemi and S. Zahediasl, "Normality Tests for Statistical Analysis: A Guide for Non-statisticians," *International Journal of Endocrinology and Metabolism*, vol. 10, pp. 486–489, 2012.

[41] B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes*. Cambridge, UK: Cambridge University Press, 1986.

[42] J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*. Milton Park, UK: Routledge, 1988.

[43] J. Clune *et al.*, "Evolving Coordinated Quadruped Gaits with the HyperNEAT Generative Encoding," in *Proceedings of the IEEE Congress on Evolutionary Computation*. Trondheim, Norway: IEEE Press, 2009, pp. 2764–2771.

[44] S. Lee *et al.*, "Evolving Gaits for Physical Robots with the HyperNEAT Generative Encoding: The Benefits of Simulation," in *Applications of Evolutionary Computation*. Vienna, Austria: Springer, 2018, pp. 540–549.

[45] L. Helms and J. Clune, "Improving HybrID: How to Best Combine Indirect and Direct Encoding in Evolutionary Algorithms," *PLoS ONE*, vol. 12, no. 3, p. e0174635, 2017.

[46] H. Lipson, "Principles of modularity, regularity, and hierarchy for scalable systems," *Journal of Biological Physics and Chemistry*, vol. 7, no. 1, pp. 125–128, 2007.

[47] P. Tonelli and J.-B. Mouret, "On the Relationships between Generative Encodings, Regularity, and Learning Abilities when Evolving Plastic Artificial Neural Networks," *PLoS ONE*, vol. 8, no. 11, pp. 99–127, 2013.

[48] A. Cohen and D. Boothe, "Sensory motor interactions during locomotion: Principles derived from biological systems," *Auton. Robots*, vol. 7, no. 3, pp. 239–245, 1999.

[49] T. Buresch *et al.*, "Effects of Evolutionary and Lifetime Learning on Minds and Bodies in an Artifical Society," in *Proceedings of the IEEE Congress on Evolutionary Computation*. Edinburgh, UK: IEEE Press, 2005, pp. 1448–1454.

[50] J. Hewland and G. Nitschke, "The Benefits of Adaptive Behavior and Morphology for Cooperation in Robot Teams," in *Proceedings of the IEEE Symposium Series on Computational Intelligence*. Cape Town, South Africa: IEEE, 2017, pp. 1047–1054.

[51] G. Nitschke and D. Howard, "Autofac: The perpetual robot machine," *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 1, pp. 2–10, 2022.