Locomotion of Boneless Creatures with Distributed Control

TIM STRAUBINGER and DAVE PAGUREK

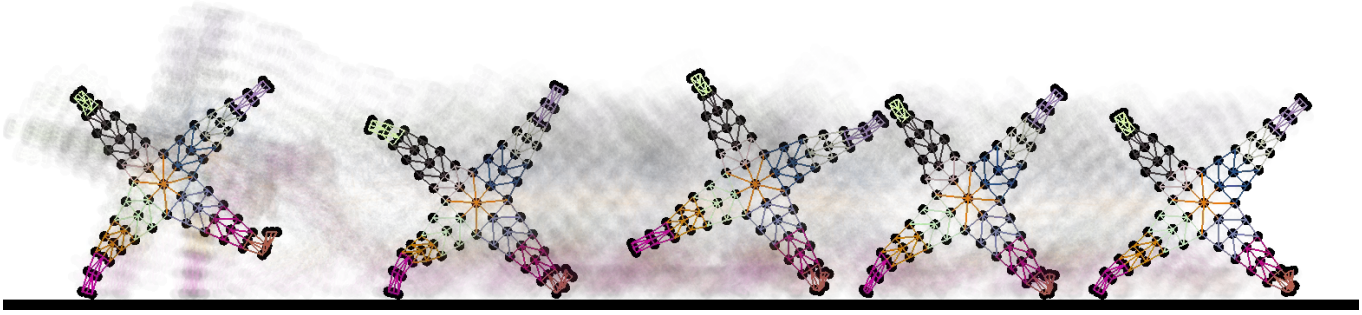

Fig. 1. A boneless tetrapus trained to walk to the right.

1 INTRODUCTION

We are interested in exploring whether we can achieve complex, organized behaviours using a distributed network of localized controllers that are tasked with collectively moving a soft body with no rigid skeleton. Given various elastic and deformable body plans, we would like to see what gaits emerge based on body shape and body material properties. Through experimentation, we comment on the effectiveness of genetic optimization over policy gradient optimization, and the expressive capability of local controllers with limited views of the world compared to single, comprehensive controllers.

2 RELATED WORK

Given a mesh, there are multiple ways one can allow a policy to control the deformation of a mesh without skeletons. One option is to allow the policy to directly deform the rest state of the mesh, independent of the elasticity model one picks [Coros et al. 2012]. Alternatively, if mesh edges are modeled as springs, a controller can change the rest length of those springs, either by finding optimal lengths given a desired centre of mass [Tan et al. 2012] or by directly controlling the rest length [Rojas et al. 2019].

The easiest way to integrate this with existing reinforcement learning algorithms is to implement our test scenario as an OpenAI Gym environment [Brockman et al. 2016]. The physics library Box2D [Catto 2020] can be used to implement the mass-spring networks and friction simulations. Box2D has been used in existing OpenAI Gym environments, making it a safe choice for implementation.

Optimization can be done using a variety of reinforcement learning algorithms, such as Proximal Policy Optimization (PPO) [Schulman et al. 2017] from the family of policy gradient algorithms, and gradient-free genetic algorithms [Such et al. 2017].

3 METHOD

3.1 Environment Setup

We implemented an OpenAI Gym [Brockman et al. 2016] environment for our problem. A policy must control the muscles of a two-dimensional soft-bodied creature and get it to walk to the right along a flat surface.

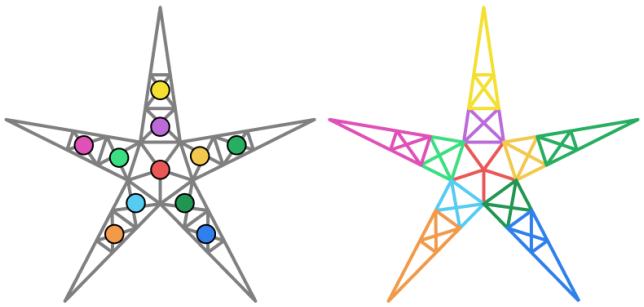

Fig. 2. Muscle group centers, shown on the left, are responsible for controlling the springs whose centers are closer to them than to other group centers, shown on the right.

The soft-bodied creature is realized as a mass-spring network in a Box2d [Catto 2020] physics simulation. The locations of the masses and springs can be imported as the vertices and faces from an OBJ file. At each vertex, a circular mass is created, where the position in configuration space for the ith mass is the point X_i. Once moved, its world coordinate is referred to as x_i. The masses are given friction with the ground and are disallowed from having any rotation, preventing them from spinning in place and thus from acting as wheels. A total mass of 10kg is assigned to the creature, distributed to masses proportional to the length of the shortest spring attached to each mass. The ith spring $S_i = \{a_i, b_i\}$ is defined by the two masses, indices a_i and b_i, that it connects. The length of the spring in configuration space is $L_i = \|X_{a_i} - X_{b_i}\|$. While the spring constants and damping constants vary between experimental setups, they are held constant throughout the entire body and within each experiment.

The creature must learn to move itself by dynamically adjusting the rest lengths of its springs. However, the policy does not directly

control spring lengths. Instead, multiple nearby springs are combined into a *muscle group*, which controls the lengths of all of its springs according to a common pattern. The jth muscle group is defined by a central point G_j in configuration space. It is responsible for controlling the springs $s_j \subseteq S$ whose configuration space center points are closest to G_j:

$$s_j = \{\{a_i, b_i\} \in S \mid \operatorname{argmin}_k \|0.5(X_{a_i} + X_{b_i}) - G_k\| = j\}. \quad (1)$$

Specifically, a muscle group will attempt to skew its springs using a linear transformation. A skew magnitude ψ, skew axis θ, and scale factor f must be provided as an action, producing the following transformation T:

$$T = \begin{bmatrix} v & \psi \\ 0 & v \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \quad v = 0.1 + \frac{1}{1+\exp(-f)}. \quad (2)$$

For each spring $i \in s_j$, a new target rest length is produced by applying T to the spring's endpoints in configuration space (i.e, their locations in the rest pose), relative to the muscle group center, and then measuring the Euclidean distance between them:

$$\tilde{l}_i = \|T(X_{a_i} - G_j) - T(X_{b_i} - G_j)\|. \quad (3)$$

The actual rest length of the spring is updated to a linear combination of its previous rest length and the desired rest length, clamped to a limited range compared to its configuration space length L_i, to add a level of smoothness:

$$l_i \leftarrow 0.5 l_i + 0.5 \min(1.2L_i, \max(0.8L_i, \tilde{l}_i)). \quad (4)$$

This aids against rapidly oscillating motion, a style which is considered to be qualitatively undesirable.

To produce ψ, θ, and f for each muscle group as an action, policies are allowed to observe the spring lengths of all springs associated with that muscle group. The policy receives no other information throughout its execution.

The reward at each time step is the body's horizontal center of mass, relative to its starting position. The episode duration was fixed within each experiment.

We tested policies that control two body plans: four-legged creature we dub the "tetrapus," and a worm, shown in Figure 3.

3.2 Policy Representations

3.2.1 Global Policy. We implement and evaluate the performance of a control policy ϕ_{global} which is both able to observe the state of the entire creature as well as control the entire creature simultaneously. This policy takes the lengths of all springs in the soft body as input and produces deformation values ψ, v, and f for every muscle group. This control policy is implemented as a neural network with one hidden layer containing 8 neurons and using the rectified linear unit (ReLU) activation function. The learnable parameters of this policy are the entries of the two weight matrices between the input, hidden, and output layers. For training with Proximal Policy Optimization (PPO), we extend this policy with random exploration by adding Gaussian noise to the outputs of the network. The standard deviation of this noise is given as an additional set of learnable parameters, specifically consisting of one scalar for each of the three deformation parameters of every muscle group. This two-layer neural network is chosen for its simplicity as well as its generality, and to establish a reasonable baseline against which to compare our distributed policy.

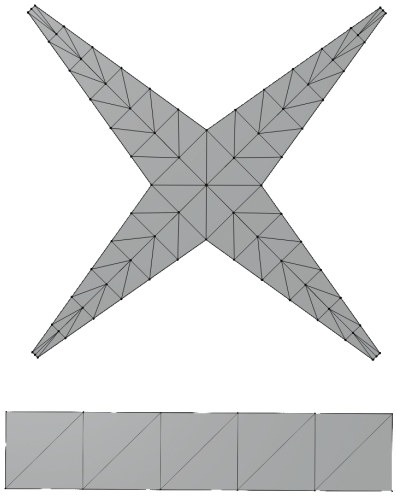

Fig. 3. Body plans controlled by our policies: tetrapus on top, worm on bottom.

3.2.2 Distributed Policy. We additionally implement and experiment with a distributed policy ϕ_{dist}, which is best thought of as a network of smaller, localized policies, rather than a single policy. The distributed control policy contains one *muscle controller* ϕ_{muscle} for each muscle group in the creature. Each such muscle controller receives as input only the lengths of those springs in its corresponding muscle group, and produces as output the deformation values ψ, θ, and f for that muscle group alone. Specifically, for the jth muscle group in the creature containing springs s_j, the input to the muscle controller ϕ_{muscle_j} is given by

$$\sigma_j = \begin{bmatrix} \vdots \\ \|x_{a_n} - x_{b_n}\| \\ \vdots \end{bmatrix}, \quad \{a_n, b_n\} \in s_j. \quad (5)$$

Similar to the global policy ϕ_{global}, this policy consists of a linear mapping to an internal latent representation x_j which is a point in $\mathbb{R}^8$. A second linear mapping produces the 3 deformation values from this latent representation. These two linear mappings constitute the learnable parameters of each muscle controller.

We add an optional extension to this policy, which we simply dub "communication," wherein each muscle controller additionally receives a weighted sum of the *previous* latent state of each muscle controller that it is immediately adjacent to. The muscle controllers adjacent to the controller ϕ_{muscle_j} are those whose muscle groups share a common mass with the controller's own muscle group, given by

$$A_j = \{k \mid (\exists\, a, b, c)[\{a, b\} \in s_j \wedge \{a, c\} \in s_k]\}. \quad (6)$$

The latent states of these controllers are summed using the controller's coefficient vector m_j as

$$y_j = \sum_{k \in A_j} m_{j,k} x_{j_{\text{prev}}}. \quad (7)$$

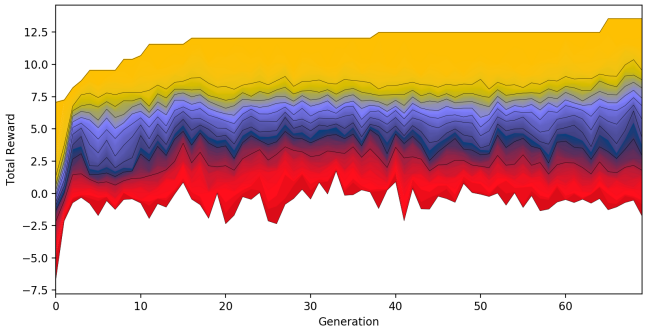

Fig. 4. Distribution of rewards per generation of tetrapodes using genetic optimization. Each reward collected in each generation comes from a different deterministic policy. Colours indicate each percentile of the population's performance. A black line is placed every 10 percentiles.

The second linear mapping is extended to accept both this sum of neighbouring previous states y_j and the original, current latent state x_j, from which the deformation parameters are then produced. This communication adds the coefficients for the weighted summation of neighbouring states as an additional learnable parameter.

During training, we found that many policies considered succesful according to our reward metric had unnaturally jerky and rapid oscillations in their movements. To avoid this, we limited the rate at which phi_{dist} receives new observations to one in every 4 frames, between which the previous deformation outputs are held constant. In our subsequent comparisons of our different policies, phi_{global} was modified similary to facilitate a more fair comparison.

3.3 Optimization Methods

Given our various policy formulations, we also want to compare the effectiveness of reinforcement learning using optimization methods from two different families: policy gradient methods and genetic methods. We optimize our policies using both the gradient-based Proximal Policy Optimization (PPO) [Schulman et al. 2017], and the gradient-free Genetic Algorithm (GA) [Such et al. 2017] with asexual reproduction and elitism.

PPO trains stochastic policies, so for this method, the network output is interpreted as the mean values of a Gaussian distribution, and the standard deviations must additionally be learnt. Standard deviations for each parameter begin at $\sigma = 0.5$ and are updated as the policy learns. We use 10 rollouts per epoch, each lasting 30 simulated seconds at 30 steps per second.

Each iteration of our GA implementation takes a population of $N = 200$ policies and performs one rollout with each. The performance of each policy is quantified by the undiscounted cumulative sum of rewards throughout the episode. The policies achieving the single best reward persists to the next generation unchanged. The remaining 199 future policies are created by randomly selecting one of the top $T = 10$ policies and then adding a random jitter $j \sim \mathcal{U}(-0.05, 0.05)$ to each parameter in the policy's neural network. The policy parameters in the first iteration of the algorithm are picked from $\mathcal{U}(-1, 1)$.

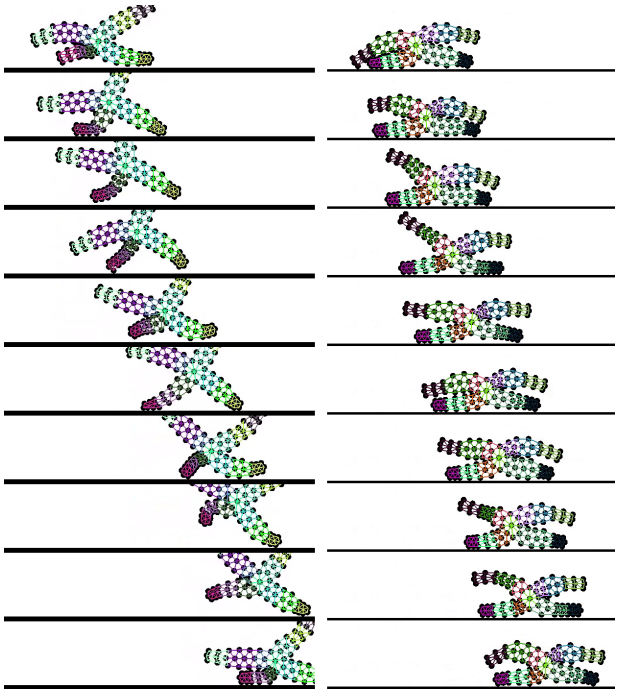

Fig. 5. Two different gaits produced using genetic optimization: one with a high spring constant on the left, and one with a low spring constant on the right.

4 RESULTS

Short animations of the results described here can be found in the readme of our project on Github[1].

4.1 Optimization Methods

After very few generations of the genetic algorithm, successful gaits are consistently discovered that take both worm and tetrapus body plans from one side of the screen to the other. This can be seen in Figure 4, which shows the distribution of returns achieved by members of the population in each generation of optimization. Curiously, the top and bottom 10 percentiles appear to be over-represented. It should be noted that the top-most line in this graph is monotonically increasing, which is guaranteed by the combination of the use of elitism (persisting the top individual without mutation) in the GA and determinism of the environment. Figure 5 shows some example gaits that are discovered when varying the springiness of the muscle material.

Policies optimized with PPO seem more limited in the gaits they discover. Movement is jittery and not as obviously periodic as is seen in the policies produced using genetic optimization. We suspect that this is due to the standard deviation in the trained stochastic policy remaining high. This can be seen in Figure 7, showing at each epoch the distribution of returns for multiple rollouts of the same stochastic policy. For the tetrapus body plan, PPO was not able to significantly reduce the variance in received rewards. Unlike the GA

[1]Codebase: https://github.com/davepagurek/boneless

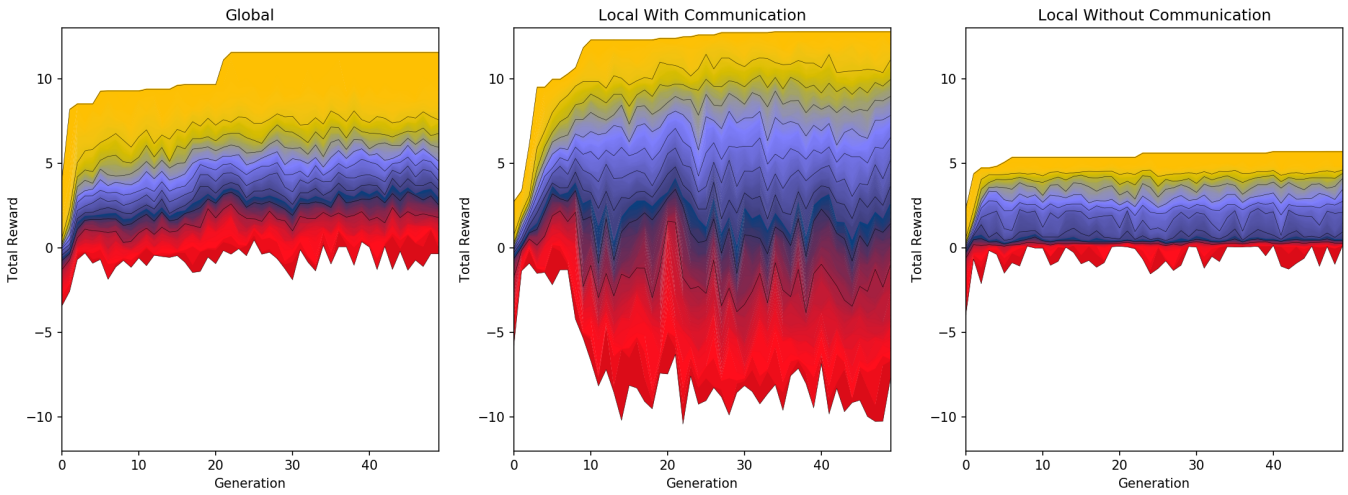

Fig. 6. Distribution of rewards per epoch given a global policy, a set of local policies with the ability to communicate with neighbouring policies, and a set of local policies without the ability to communicate. In this experiment, the monolithic policy ϕ_{global} was modified to receive new state only every 4 frames, similarly to ϕ_{dist}.

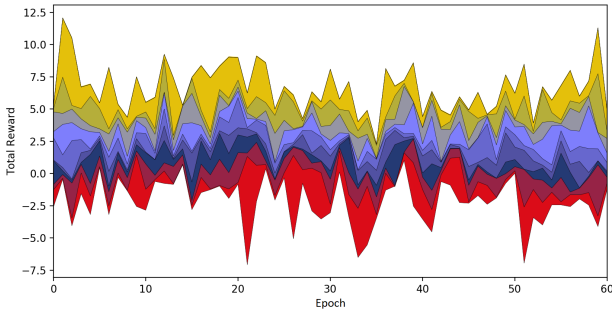

Fig. 7. Distribution of rewards per epoch of PPO optimization of the tetrapus. Note that for a given epoch, all rewards are collected from multiple rollouts of the same stochastic policy.

approach, PPO requires stochasticity in its policies for exploration, but in a similar number of rollouts, it is not able to refine its policy enough to *consistently* achieve high rewards.

4.2 Local Policy Control

To compare how effective distributed policy control can be, we optimized three types of policies for the worm body plan. First, we optimized the monolothic policy with full knowledge and control, ϕ_{global}. Next, we optimized a set of distributed policies ϕ_{dist} with communication. Finally, we also optimized a set of distributed policies ϕ_{dist} without communication. Figure 6 shows the distribution of rewards across fifty generations of GA optimization using each control scheme.

Both the global scheme and the local communication scheme are able to create policies that achieve high rewards. This indicates that distributed controllers are expressive enough to learn complex movements. The removal of the ability of local policies to communicate significantly reduces the maximum achievable reward, indicating that communication is essential if local policies are to effectively work together to towards a goal.

We also note that the gaits demonstrated in the final generation of this experiment are fairly similar to the gaits only a few generations in. Effective gaits are quickly discovered and remain stable throughout the rest of the optimization. We acknowledge that these results may not be due to the differences in policy designs alone and may also be in part accounted for by different random initializations resulting in better policies. Understanding this better would add to our understanding of our distributed policy and the GA algorithm, but would also require far greater computational resources.

5 FUTURE WORK

While the high standard deviation contributed to the weakness of PPO for this task, we would like to see if this can be addressed. A behaviour characteristic of a stochastic PPO-trained policy is its jitteriness. This can perhaps be discouraged, for example, by subtracting from the environment's rewards when the creature experiences high levels of jerk, aiming to encourage smoothness in the policy.

We observed that the gaits which were found in each run of GA seemed to be largely determined during the first, randomly initialized population of policies. We did not observe major qualitative changes in locomotion style in subsequent generations. This suggests that our GA implementation may be limiting itself to local minima in the policy space, and it would be preferable to be able to reliably explore a larger variety of successful policies. Pursuing this will likely require much larger population sizes and possibly other well-documented extensions to the GA.

We tested our policies on a flat terrain for between 10 and 30 simulated seconds. Future experiments could train policies on varying terrain for longer periods of time to try to achieve robustness. This

robustness can then be evaluated using the classic task of walking while being pelted from all angles with cubes.

Finally, with our experiment infrastructure in place, we would like to run experiments on a variety of other body plans. Our method may be extended to allow the physical shape and material parameters of the soft-bodied creature to be evolved in addition to its control policy, and performing experiments with this approach would likely allow us to evolve creatures that are physically better suited to their environments and may yield more surprising and insightful physical body plans and methods of locomotion that would readily not occur to us as designers.

REFERENCES

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. arXiv:arXiv:1606.01540

Erin Catto. 2020. Box2D. https://box2d.org/.

Stelian Coros, Sebastian Martin, Bernhard Thomaszewski, Christian Schumacher, Robert Sumner, and Markus Gross. 2012. Deformable Objects Alive! *ACM Trans. Graph.* 31, 4, Article Article 69 (July 2012), 9 pages. https://doi.org/10.1145/2185520.2185565

Junior Rojas, Stelian Coros, and Ladislav Kavan. 2019. Deep reinforcement learning for 2D soft bodylocomotion. In *Conference on Neural Information Processing Systems*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. (07 2017).

Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. 2017. Deep Neuroevolution: Genetic Algorithms Are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning. arXiv:cs.NE/1712.06567

Jie Tan, Greg Turk, and C. Karen Liu. 2012. Soft Body Locomotion. *ACM Trans. Graph.* 31, 4, Article Article 26 (July 2012), 11 pages. https://doi.org/10.1145/2185520.2185522