Can Chaos be utilized as exploration noise for locomotion learning?

Worasuchad Haomachai[1] and Poramate Manoonpong[1,2*]
[1]Nanjing University of Aeronautics and Astronautics, Nanjing, China
[2]Vidyasirimedhi Institute of Science & Technology, Rayong, Thailand
*haomachai@gmail.com, *poramate.m@vistec.ac.th*

1 Introduction

There is compelling evidence to support that chaotic patterns of behavior exist in many biological systems [1,2]. For example, Maye et al. [2] observed behavioral indeterminacy (comparable to a chaotic pattern) during spontaneous flight maneuvers (searching behavior without any external cues) in Drosophila fruit flies. This suggests that chaotic dynamics may be involved in the biological neural control underlying spontaneous behavior. It also raises the question, "Can chaos be utilized in artificial neural control for robot locomotion learning?" To address the question, this study investigates and compares the use of chaotic exploration noise and standard Gaussian noise for robot locomotion learning. Although chaos has been used to tackle machine learning problems (such as classification) [3], until now, it is yet to be thoroughly explored for locomotion learning.

2 Materials and Methods

For our investigation, we construct a locomotion controller as a reinforcement learning framework, so that our robot (here, a gecko-like robot) has to learn to walk. The controller, configured as a neural central pattern generator (CPG) with a radial basis function (RBF)-based premotor neuron network (Fig. 1), is based on our previous work [4]. In this controller, the robot joint trajectories are encoded in the output weights connecting the CPG-RBF network to the motor neurons (dashed lines in Fig. 1). The output weights are learned using a probability-based black-box optimization (BBO) approach to optimize joint trajectories with respect to robot walking performance.

Our simulated gecko-like robot consists of a bendable body with three active joints for lateral body undulation and four identical legs, each with four active joints for leg forward/backward motion ($j1$, Fig. 1), leg elevation/depression ($j2$, Fig. 1), foot attachment/detachment ($j3$, Fig. 1), and leg flexion/extension ($j4$, Fig. 1). The body joints and leg joints are controlled separately. For simplicity, we employ a CPG signal with predefined connections to drive the three body joints ($b1, b2, b3$) to achieve a C-shaped standing wave pattern (see Fig. **??**, left), while the trajectories or patterns of all 16 leg joints ($j1, j2, .., j16$) are optimized by BBO.

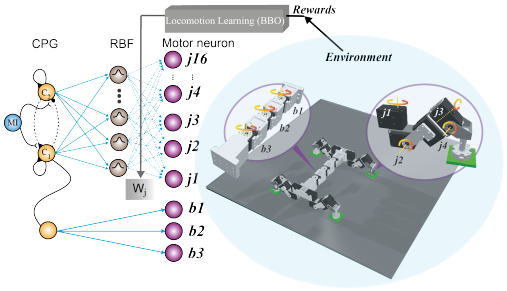

Figure 1: An overview of the CPG-RBF control network with BBO applied to the gecko-like robot. The robot simulator CoppeliaSim and physics Vortex are used for robot simulation.

2.1 Locomotion Control Network

The whole network consists of three layers. For the input layer, a CPG based on a two-neuron recurrent network is used to produce two sinusoidal-like signals. These CPG output signals are fed into the hidden layer which consists of 20 RBF neurons. The RBF neurons are equally distributed along one period of the CPG output signals. Subsequently, RBF outputs are projected to all motor neurons for controlling robot motor joint positions. According to the setup, the network's output weights can be optimized via BBO to shape and translate the original CPG sinusoidal-like signals into final complex joint trajectories that yield great robot walking performance.

2.2 Locomotion Learning

Algorithm 1 BBO

while *cost not converged* **do**
  for $k \in K$ **do**
    $R_k = execCPGRBFN(w_{k,j} + \varepsilon_k)$
  end
  for $k \in K$ **do**
    $S_k = e^{\lambda \cdot \frac{R_k - \min(R)}{\max(R) - \min(R)}}$
    $P_k = \frac{S_k}{\sum_{k=1}^{K} S_k}$
  end
  $\delta w_{k,j} = \sum_{k=1}^{K} P_k \cdot \varepsilon_k$
  $w_{k,j} \leftarrow w_{k,j} + \delta w_{k,j}$
  $\varepsilon_k \leftarrow \gamma \cdot \varepsilon_k$
end

The BBO method, used here called PI^{BB}, is a parameter perturbation approach and a variant of an evolutionary algorithm. It is employed to optimize the output weights in order to generate optimal joint trajectories that maximize a reward, which basically describes how well the robot performs (Fig. 2A). The pseudocode of BBO can be seen in Algorithm 1. BBO is a probability-based ap-

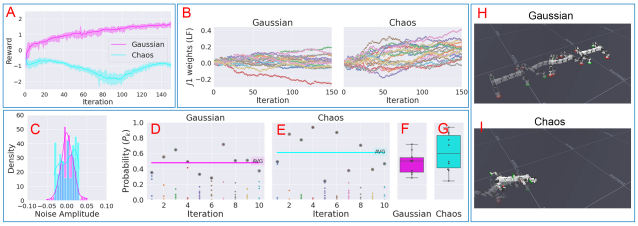

Figure 2: Control parameter optimization by Gaussian and chaotic noise. (A) Reward signals. (B) 20 output weights, projecting to the joint 1 ($j1$) of left front leg (LF), optimized by Gaussian and chaotic noise. (C) Gaussian (pink) and chaotic (blue) noise distribution profiles. (D), (E) The first ten iterations of P_k and their highest value distribution (F), (G), optimized by Gaussian and chaotic noise. (H), (I) The robot locomotion performance after learning 150 iterations. A video of this can be seen at http://www.manoonpong.com/AMAM2023/BBO/Video.mp4

proach where noise ε is added to the weights $\mathbf{w}$ in order to explore the parameter space. Gaussian noise $\mathcal{N}(0,\sigma^2)$ is typically used. After the noise is added, the weights are applied to generate the joint trajectories and robot locomotion is evaluated through a reward function R. In this study, we use K roll-outs ($K=16$) with different noise ε_k to produce new weight sets $\mathbf{w}_{k,j}$ and obtain the rewards R_k. The probability for each roll-out P_k is calculated using cost-weighted averaging in order to update the new weights. The perturbation noise is further linearly decayed using a decay constant of $\gamma = 0.995$. The reward function is for the robot moving forward in a straight line as specified by $R_k = 3 \cdot distance - (instability + 0.1 \cdot slipping + collision + 0.2 \cdot power)$, where distance is a measure of how far the robot has traveled in a straight line. The penalty term consists of the following components. Instability is a measure of how stable the robot is during locomotion; collision is a measure of the extent to which one leg of the robot collides with another; slipping is a measure of the extent to which each leg of the robot slips on the ground; and power is the energy consumed during traversing.

3 Experiment and Results

Inspired by biological systems, we aim to determine whether chaotic noise can be utilized as a perturbation noise to optimize control parameters (output weights) in BBO. Thus, we let the robot learn with chaotic noise (ε_k = chaotic noise) and compared its locomotion learning performance to Gaussian noise (ε_k = Gaussian noise with $\sigma^2 = 0.015$). Note that the chaotic noise was generated using a chaotic two-neuron recurrent network. In both cases, the learning period was set to 150 iterations with 15 s of simulation time per iteration. A comparison of the chaotic and Gaussian noise distribution profiles is shown in Fig. 2C. As can be observed, the chaotic noise shows an asymmetric profile and the Gaussian noise a symmetric one. Due to the asymmetric profile and chaotic dynamics, the control parameters might change quickly at the beginning, subsequently converting to a certain parameter space, as observed in the weight changes of $j1$ of LF (Fig. 2B). They quickly increased and then persisted in the same positive direction. Based on the analysis of BBO during the first ten iterations, we observed that the highest probability P_k of chaotic noise tends to fluctuate and dominate by performing undesirable behaviors (Fig. 2E). The average value of the highest P_k in each iteration was 0.60 with an SD of 0.27 (Fig. 2G). Undesirable behaviors that dominate at the beginning can quickly lead the optimization process in the wrong direction (see video). As a consequence, the optimization process could get stuck at the local optima, preventing the robot from forming a stable gait for walking forward (Fig. 2I, right, and the section below for short explanation). In contrast, the symmetric profile of Gaussian noise can slowly adapt the parameters, leading to a balance of positive and negative parameter values with lower probability and a variant of P_k (Fig. 2D and 2F). The average of the highest P_k in each iteration was 0.47 with an SD of 0.13 (Fig. 2F). This results in preventing divergence (Fig. 2B) where a stable gait can be formed (Fig. 2H).

4 Discussion and Conclusion

Our investigation reveals that chaos cannot be directly utilized as exploration noise in BBO for locomotion learning. This is due to its asymmetrical distribution profile which quickly adapts control parameters but creates an imbalance between positive and negative value exploration. In other words, the parameters will be quickly updated, potentially overshooting an optimal solution; thereby failing to converge to a solution or becoming stuck at local minima. On the other hand, the Gaussian symmetric distribution profile slowly updates the control parameters that can converge to an optimal solution, enabling the robot to successfully walk. Although chaotic noise fails for locomotion learning here, it seems to facilitate learning speed (i.e., it can adapt parameters faster than Gaussian noise). Therefore, we will further explore an alternative strategy that uses chaotic dynamics to accelerate the overall optimization process of BBO with Gaussian noise for fast and stable locomotion learning.

5 Acknowledgments

This work was supported by the National Key R&D Program of China (2020YFB1313504)[PM]. We thank the CM labs for providing Vortex.

References

[1] Faure P, Korn H (2001) Is there chaos in the brain? I. Concepts of nonlinear dynamics and methods of investigation. C R Acad Sci III.

[2] Maye A, Hsieh CH, Sugihara G, Brembs B. Order in spontaneous behavior. PloS one, 2(5):e443.

[3] Harikrishnan NB, Nagaraj N (2021) When noise meets chaos: Stochastic resonance in neurochaos learning. Neural Networks.

[4] M. Thor, T. Kulvicius and P. Manoonpong, "Generic Neural Locomotion Control Framework for Legged Robots," in IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 9, pp. 4013-4025, Sept. 2021.