AU 332 Artificial Intelligence: Principles and Techniques

By: Shuo Yang and Zihao Li

Project Report

January 22, 2020

I. INTRODUCTION

A. Motivation

Learning policies for bipedal locomotion can be difficult, as experiments are expensive and simulation does not usually transfer well to hardware. We typically do not have robots that can take a fall, and it is cumbersome to perform these experiments. To counter this, we need algorithms that are sample efficient. On top of this, most objective functions are non-convex, non-differentiable and noisy. With these considerations in mind, it is important to find optimization methods that are sample efficient, robust to noise and non-convexity, and try to minimize the number of bad policies sampled.

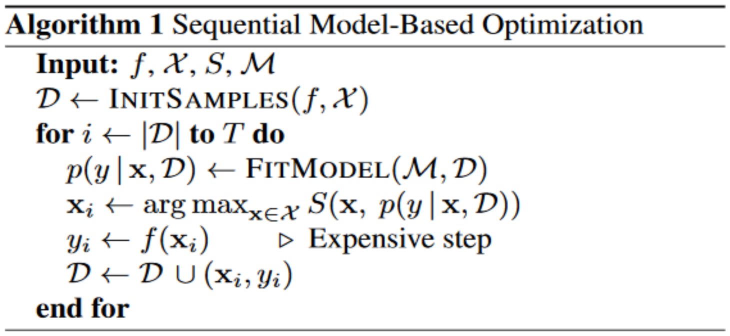
B. Background

1. Central Pattern Generator

Existing neurobiological studies have revealed that rhythmic motor patterns are controlled by neural oscillators referred as central pattern generators (CPGs), and this CPG mechanism is good for both adaptability and stability of animals. See, e.g., [7], [8]. Among these studies, a human-like biped walking was successfully simulated by using a CPG controller.

2. Bayesian optimization

Bayesian optimization (BO) is a global optimization method based on response surface (i.e., surrogate model). Bayesian optimization has been re-discovered multiple times by different communities and is also referred to as efficient global optimization (EGO) and sequential kriging optimization (SKO). See, e.g., [3], [10], [9], [16]. The algorithm of BO is shown below.



Algorithm 1 Sequential Model-Based Optimization

```
Input: $f, \mathcal{X}, S, \mathcal{M}$
$\mathcal{D} \leftarrow$ INITSAMPLES($f, \mathcal{X}$)
for $i \leftarrow |\mathcal{D}|$ to $T$ do
    $p(y \mid \mathbf{x}, \mathcal{D}) \leftarrow$ FITMODEL($\mathcal{M}, \mathcal{D}$)
    $\mathbf{x}_i \leftarrow \arg\max_{\mathbf{x} \in \mathcal{X}} S(\mathbf{x},\ p(y \mid \mathbf{x}, \mathcal{D}))$
    $y_i \leftarrow f(\mathbf{x}_i)$        ▷ Expensive step
    $\mathcal{D} \leftarrow \mathcal{D} \cup (\mathbf{x}_i, y_i)$
end for
```

C. Related Works

Robot locomotion has drawn many attentions recently in the literature, see, e.g., [8], [14], [11], [6], [17], [15] [1]. Some works focus on sample efficient optimization for biped locomotion, see, e.g., [2], which is similar to our work.

D. Equipment

There is a minimal amount of equipment to be used in this project. The few requirements are listed below:

- JetBrains PyCharm Community Edition 2019.2.1 x64
- Jupyter
- Laptops capable of running the software mentioned
- IDLE(Python 3.6)

E. Procedure

1. Have some meetings with Prof. Gao and TA Ming Sun, then choose the main direction of our project.
2. List our project plan.
3. Modify CPG codes.
4. Implement random search, genetic algorithms and bayesian optimization in the framework of CPG, and then train NAO in V-REP.
5. Transfer policy trained in V-REP to real physical NAO.
6. Give a presentation and write report.

F. Organization

The rest of this report is organized as follows. In Section II, we introduce the generation of biped locomotion patterns, performance evaluation and some algorithms of this project in detail. Then we introduce our experiment setup and results in Section III. In Section IV, we conclude this report and give our acknowledgement.

II. MEHTODS & PERFORMANCE

A. Generation of Biped Locomotion Patterns

We designed a biped motion controller based on CPG network previously proposed by Julian Christiano in [4]. We use CPG network to generate rythmic signal as the target angle for each of the robot's joints in order to describe a valid locomotion pattern.

1. Basic Oscillator

The CPG network is composed of a set of interconnected nonlinear oscillators previously proposed by Matsuoka in [12] and [13]. Each non-linear oscillator consists of two neurons with a self-inhibition effect, which are reciprocally linked via inhibitory connections. The behavior of each neuron is described as follows.

$$\begin{cases} \tau \dot{u_{1i}} = -u_{1i} - \omega_0 y_{2i} - \beta v_{1i} + u_e + f_{1i} + a s_{1i}, \\ \tau' \dot{v_{1i}} = -v_{1i} + y_{1i}, \\ y_{1i} = max(0, u_{1i}), \end{cases} \tag{1}$$

$$\begin{cases} \tau \dot{u_{2i}} = -u_{2i} - \omega_0 y_{1i} - \beta v_{2i} + u_e + f_{2i} + a s_{2i}, \\ \tau' \dot{v_{2i}} = -v_{2i} + y_{2i}, \\ y_{2i} = max(0, u_{2i}), \end{cases} \tag{2}$$

In which subscript 1 corresponds to the extensor neuron and subscript 2 corresponds to the flexor neuron. Each oscillator has four state variables: $u_{1i}, v_{1i}, u_{2i}, v_{2i}$. u_e is an external input which affects the amplitude of the oscillator output. f_{1i} and f_{2i} are feedback variables which can be used to control the output amplitude and phase. Parameter s_{1i} and s_{2i} are defined as below.

$$\begin{cases} s_{1i} = \Sigma \omega_{ij} u_{1j}, \\ s_{2i} = \Sigma \omega_{ij} u_{2j}, \end{cases} \tag{3}$$

where ω_{ij} represents the unidirectional connection weight between a master oscillator O_j and a slave oscillator O_i. The output signals of the oscillators are calculated as below:

$$o_i = -m_1 y_{1i} + m_2 y_{2i}$$

m_1 and m_2 control the amplitude of the output signal. In order to modulate the frequency of the oscillator a parameter k_f is introduced in [4], the time constants are reformulated as:

$$\begin{cases} \tau = \tau_0 k_f, \\ \tau_0' = \tau_0' k_f, \end{cases} \tag{4}$$

Because the frequency would greatly affect the gait of the hardware robot and the reduction in dimensionality is beneficial to reducing training time. We set the parameter k_f to a certain value of 0.462 in this work.

2. CPG Network

The CPG network follows a master-slave topology, in which a central non-linear oscillator is used to drive the different joints of the robot through slave oscillators associated with each of the joints. The central oscillator works as a pacemaker adjusting the frequency of robot actions. It should be noted that we need to set the initial value of the parameters y_{1i} and y_{2i} to be negative to trigger the central oscillator, ortherwise it will generate constant signal.

The structure of the network is shown in 1. Besides the motion of legs we also take into consideration the motion of arms, which is believed to be helpful to yield a more stable walking pattern. In order to mitigate the adverse effect of high dimensionality on training time, we configured all oscillators with the same parameters except for the feedback parameter k. k is used as the factor of f_{12},f_{22},f_{13} and f_{23}. The four variables are calculated as below:

$$\begin{cases} f_{12} = k\theta_L, \\ f_{22} = -k\theta_L, \\ f_{13} = k\theta_R, \\ f_{23} = -k\theta_R, \end{cases} \tag{5}$$

where θ_L is the observed angle of LHipPitch and θ_R is that for RHipPitch. The 10 CPG network parameters: six gains(GAIN1,GAIN2,GAIN3,GAIN4,GAIN5,GAIN6) and four offsets(BIAS1,BIAS2,BIAS3,BIAS4) are also need to be tuned.

B. Performance Evaluation

We evaluate the gait by how far the robot can travel in a finite amount of time. The reward function is

$$r = k_x * (x_T - x_0) + k_y * |y_T - y_0| + k_p * isFall,$$

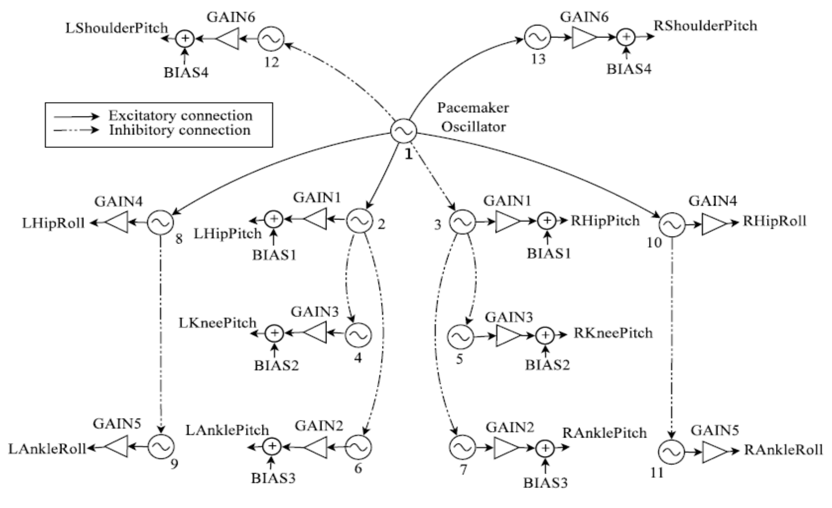


FIG. 1: Topology for the CPG network

where x is the coordinate of the midpoint of both feet on the X-axis in absolute coordinates and y is that on the Y-axis in absolute coordinates. x_0, y_0 correspond to the initial position and x_T, y_T correspond to the final position. $isFall$ indicates whether the robot falls before the simulation ends. The reason why we choose the midpoint of both feet rather than the center of gravity as reference point is that the back and forth swing will have great impact on the measurement of x_T when the simulation time is compressed in the pursuit of a fast training speed. In this study, the parameters were chosen as $k_x = 1, k_y = -0.2, k_p = -0.5$.

a. qualified parameter For a set of parameter, if the robot gains a reward higher than the standard line we set with this set of parameter then we call it a qualified parameter.

C. Algorithm

1. Bayesian Optimization

Bayesian optimization is an approach to optimizing objective functions that take a long time to evaluate. It is best-suited for optimization over continuous domains of less than 20 dimensions, and tolerates stochastic noise in function evaluations. It builds a surrogate for the objective and quantifies the uncertainty in that surrogate using a Bayesian machine learning technique, Gaussian process regression, and then uses an acquisition function defined from this surrogate to decide where to sample.

Because our goal is to find as many qualified parameters as possible, we choose UCB(upper confidence bound) as the acquisition function. UCB does not evaluate based on the points with largest value found so far, so we can find more different qualified parameters with rewards slightly higher than the standard line.

a. upper confidence bound AC function

$$UCB = \mu(x) + k\sigma(x)$$

The parameter k determines the degree to which we rely on uncertainty in choosing the next probe point.

Before probing with acquisition function, we did 300 random searches to initialize our model. To avoid probing around a suboptimal point for a long time or doing so much exploration that we ignore those areas with high-value points, we adjust parameter k dynamically. We take k equal to 1 for 50 simulations and 10 for 50 simulations alternately. By setting k equal to 10, we want the optimizer to probe the area that has not been adequately detected to find the area with potentially high value points. By setting k equal to 1, we want the optimizer to probe around the area with high expected value in case the value of the detected point in this area is very close but less than our standard line.

2. Genetic Algorithm

Genetic algorithm (GA) optimizers are robust, stochastic search methods modeled on the principles and concepts of natural selection and evolution. For more details of GA, see, e.g., [5].
In our project, we let generation be 50 and population size be 30. Finally, results show that in the former 40 generations, GA does not work well. However, after about 40 generations, GA performs better, which means that it does increase the sample efficiency.

3. Random Search

It is believed that random search works more efficiently than grid search and it is a very simple search method. So we take the results of random search as the baseline.

III. EXPERIMENT

A. Robot

We choose NAO as our experiment platform. NAO is a bipedal robot with pleasantly rounded features and it has 25 degrees of freedom which enable it to move and adapt to the environment.

B. Simulator Setup

We use simulator Vrep to do numerical simulation. We set each simulation step to run for 10 milliseconds and each simulation contains at most 600 simulation steps. If the robot falls down, the simulation will terminate immediately.

C. Simulation Results

First, we carried out numerical simulations on the simulator Vrep to acquire parameters for the CPG for biped walking. Then we implemented the acquired CPG controller on the hardware robot.

By observing the gait in the simulator, we find that if the robot can walk steadily, it will at least gain a reward around 0.07 and if the robot walks with a better gait, it will gain a reward higher than 0.1. When we do experiment on the hardware robot, falling down will make much damage to the robot, so we should also consider the falling down situation, which could be reflected by the reward less than -0.1. We set three standard lines as -0.1,0.07 and 0.1 and count how many times the reward reached these standard lines in a total of 1200 simulations for every algorithm.

For the first 300 simulations, BO and random search performs the same because we initialize BO with 300 random search. After 300 simulations, BO performs significantly better. Obviously, genetic algorithm is not an efficient searching method. We can find that for the first 1200 simulations, none of the parameters generated by GA can get a reward higher than 0.1 although after 1200 iterations the highest reward in a generation goes up dramatically.

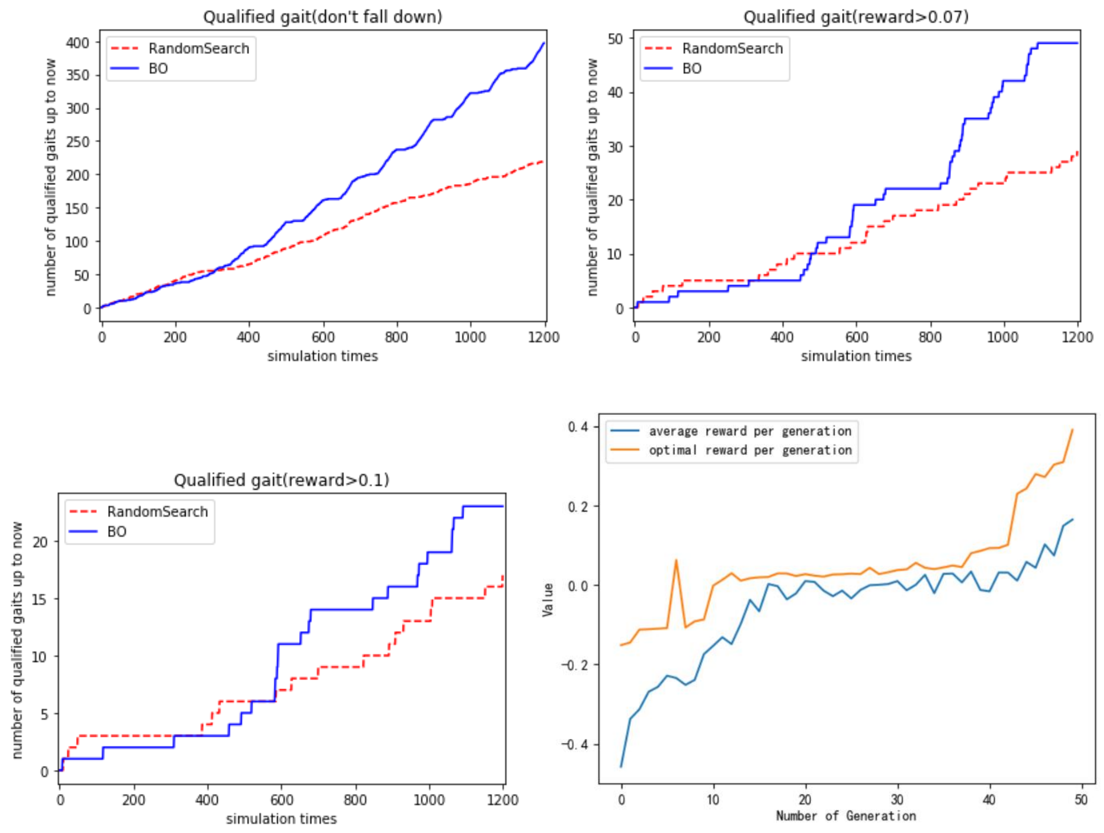


FIG. 2: Comparison of different algorithms with reference to standard lines

IV. CONCLUSION & ACKNOWLEDGEMENT

A. Conclusion & Discussion

We adapted the CPG network proposed by Julian Cristiano to control the locomotion of a biped robot. We compare Bayesian optimization with the widely used genetic algorithms on searching parameters. Experimental results shows that BO is obviously better than GA on finding qualified parameters in a given number of trials. In other words, we can find well-behaved parameters for further work more effectively with Bayesian optimization.

Future work will include tuning parameters of the oscillators to improve the gait and use visual information as feedback to improve adaptability to the environment.

B. Acknowledgement

Thanks for the help from Prof. Yue Gao and TAs. They contribute a lot to our project.

[1] R. Antonova, A. Rai, and C.G Atkeson. Deep kernels for optimizing locomotion controllers. *arXiv preprint arXiv:1707.09062*, 2017.

[2] Rika Antonova, Akshara Rai, and Christopher G Atkeson. Sample efficient optimization for learning controllers for bipedal locomotion. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 22–28. IEEE, 2016.

[3] E. Brochu, V.M Cora, and N. De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[4] Julián Cristiano, Domenec Puig, and Miguel Angel García. Locomotion control of a biped robot through a feedback cpg network. In *ROBOT2013: First Iberian Robotics Conference*, pages 527–540. Springer, 2014.

[5] Lawrence Davis. Handbook of genetic algorithms. 1991.

[6] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng. Learning cpg sensory feedback with policy gradient for biped locomotion for a full-body humanoid. In *Proceedings of the national conference on artificial intelligence*, volume 20, page 1267. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

[7] S. Grillner, P.r Wallen, L. Brodin, and A. Lansner. Neuronal network generating locomotor behavior in lamprey: circuitry, transmitters, membrane properties, and simulation. *Annual review of neuroscience*, 14(1):169–199, 1991.

[8] A.J. Ijspeert. Central pattern generators for locomotion control in animals and robots: a review. *Neural networks*, 21(4):642–653, 2008.

[9] D.R Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21(4):345–383, 2001.

[10] H.J Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86(1):97–106, 1964.

[11] T. Liao, G. Wang, B. Yang, R. Lee, K. Pister, S. Levine, and R. Calandra. Data-efficient learning of morphology and controller for a microrobot. *arXiv preprint arXiv:1905.01334*, 2019.

[12] Kiyotoshi Matsuoka. Sustained oscillations generated by mutually inhibiting neurons with adaptation. *Biological cybernetics*, 52(6):367–376, 1985.

[13] Kiyotoshi Matsuoka. Mechanisms of frequency and pattern control in the neural rhythm generators. *Biological cybernetics*, 56(5-6):345–353, 1987.

[14] T. Mori, Y. Nakamura, M.-A. Sato, and S. Ishii. Reinforcement learning for cpg-driven biped robot. In *AAAI*, volume 4, pages 623–630, 2004.

[15] J. Morimoto, G. Cheng, C.G Atkeson, and G. Zeglin. A simple reinforcement learning algorithm for biped walking. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 3, pages 3030–3035. IEEE, 2004.

[16] M.A Osborne, R. Garnett, and S.J Roberts. Gaussian processes for global optimization. In *3rd international conference on learning and intelligent optimization (LION3)*, volume 2009, 2009.

[17] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.