Author Proof

Bayesian Optimization of a Quadruped Robot During 3-Dimensional Locomotion

Jiahui Zhu[1(✉)], Shuting Li[1,2], Zhuoli Wang[1], and Andre Rosendo[1]

[1] Living Machines Lab, School of Information Science and Technology, ShanghaiTech University, Shanghai, China
{zhujh1,wangzhl,arosendo}@shanghaitech.edu.cn
[2] College of Electronics and Information Engineering, Sichuan University, Chengdu, Sichuan, China
lishuting.cd@gmail.com
http://lima.sist.shanghaitech.edu.cn/

Abstract. Parametric search for gait controllers is a key challenge in quadruped locomotion. Several optimization methods can be adopted to find the optimal solution by regarding it as an optimization problem. Here we adopt Bayesian optimization (BO), a global optimization method that is suitable for unknown objective functions particularly when it is hard to evaluate, which is the common case of real robot experiments. We demonstrate this process on a quadruped robot capable of 3-dimensional locomotion, and our goal is to make it move forward as far as possible. While initially probing the parametric landscape, Random Search shows that in a 10-dimensional search space of over a million combinations, only 30% of them contribute to moving forward, merely 2% results in our robot walking longer than 2 m, and none of these parameters leads to more than 3 m distance. In face of such difficult landscape BO finds near-optimal parameters after 22 iterations, and walks a range of 3 m in over 40% of its iterations. Our findings illustrate that BO can efficiently search control parameters in a 3-dimensional locomotion case, and the development of controllers for legged robots, very often plagued with manual tuning of parameters, could profit from this.

Keywords: Gait optimization · Quadruped locomotion · 3-dimensional locomotion

1 Introduction

Legged robots overcome several limitations of traditional wheeled robots [1]. From all forms of legged robots, quadruped robots outperform one-legged and bipedal robots in stability and robustness, and possess higher mobility and versatility in walking, running, and jumping, than hexapod and octopod due to bionic reasons. Despite many advances in design and control of four-legged robots [2], the walking gait of quadruped locomotion is still fundamental yet challenging.

© Springer Nature Switzerland AG 2019
U. Martinez-Hernandez et al. (Eds.): Living Machines 2019, LNAI 11556, pp. 1–12, 2019.
https://doi.org/10.1007/978-3-030-24741-6_26

Author Proof

Some fitting controllers for quadruped robots have already been designed [3], but the real problem then is the parameter set-up. The performance of such controller parameters largely depends on the surrounding environment, which is unpredictable and noisy. In that case, parametric search becomes a dynamic problem and often requires artificial interactions. At the beginning, gait optimization often transfers to a trial-and-error algorithm [4], yet robots may break easily if obtained parameters perform badly. Consequently, manual tuning of parameters relies on expertise in engineering. Moreover, a variation in walking surface (e.g., angle, friction, and hardness), a change in optimization target (e.g., velocity, energy efficiency, and stability), can result in a new search of parameters. In conclusion, a quickly adaptable automatic search is in urgent need.

By considering the search for gait parameters as an optimization problem, multiple optimization techniques can be applied to find the local or global extreme. Some of the recent approaches, particularly in robotics, mainly include: gradient descent [5], genetic algorithms [6–8], and Bayesian optimization [9–13], where the first two methods each have major drawbacks. While the gradient descent method requires a known objective function as well as access to its derivatives, which are both hard to obtain in robotics experiments, genetic algorithms is time-consuming or even unfeasible when using fragile robots, requiring many generations to converge [6].

BO is a novel strategy to black-box global optimization, especially when the objective function is expensive to evaluate [14]. It uses Gaussian Processes to approximate the (true) objective function and an acquisition function to promote a balance between exploitation and exploration in the selection of the next iteration [9]. In [10,11], and [12] BO is applied in a 2-dimensional bipedal locomotion problem, where [10] is used in simulations and [11,12] being applied to a real bipedal robot. In [13] the authors show that a hexapod robot could use BO to adapt their controllers to damage.

In this article we implement BO on an open-loop quadruped robot with 3-dimensional locomotion to effectively find a near-optimal set of parameters for the controller, which is selected from a search space of 1048576 ($\approx 10^6$) combinations. We present the Methods used in this work at Sect. 2, including Gaussian Processes, Bayesian optimization, Random Search, hardware set-up, and parameter configuration. In Sect. 3, we demonstrate experimental results with several figures and further discussion, while we conclude our work in the last section.

2 Method

2.1 Gait Optimization Methods

The work of [11] suggests that the parametric search of a controller can be viewed as an optimization problem, which can be formulated as follows:

$$\max_{x \in \mathbb{R}^d} f(x) \tag{1}$$

Author Proof

In the realm of robotics, $f(\cdot)$ is the objective function that encodes measurement standard and in our work, it is the vertical displacement between initial and end position. Meanwhile, x are the parameters that need to be tuned.

Gaussian Processes. The fundamental work of Gaussian processes (GPs) is introduced in [15]. A Gaussian process can be defined as a probabilistic nonlinear regression and provide a method for modeling probability distributions over functions, f with mean function, m and covariance function, k:

$$f(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot)) \tag{2}$$

Generally speaking, any real-valued function $m(\cdot)$ can be accepted, but it is more demanding for covariance function $k(\cdot, \cdot)$, where the kernel matrix is given by (a set of elements $x_1,\ x_2,\ ...,\ x_m \in \chi$):

$$K = \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{bmatrix}$$

Kernel Functions. Here we introduce the *Matérn* kernel [15], adopted in this work and represented by:

$$k_{Matern}(d) = \sigma_p^2 \frac{2^{1-v}}{\Gamma(\nu)} \frac{\sqrt{2\nu d}^{v}}{l} K_\nu \frac{\sqrt{2\nu d}}{l} \tag{3}$$

where d is the Euclidean distance between two inputs, $\Gamma(\nu)$ and K_ν denotes the Gamma function and the Bessel function of order ν, and l shows the characteristic length-scale. It also takes noise into account, which is denoted by σ_p^2. *Matérn* kernel has higher flexibility for both smooth and sharp areas [16].

Gaussian Process Regression. Gaussian Processes provide us a powerful tool to parameterize probability distributions over functions [15]. Consider how GPs are used in regression. Suppose that we have drawn our data, both the training and test sets, from a zero-mean prior Gaussian distribution,

$$f(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot)) \tag{4}$$

and decompose K as: $\begin{bmatrix} K & K_* \\ K_*^T & K_{**} \end{bmatrix}$, where star (*) marks for training sets, then the conditional distribution of the latent function $f(\cdot)$ can be written as:

$$y_* | ((x_1, y_1), ..., (x_n, y_n), x_*) \sim \mathcal{N}(K_*^T K^{-1} y, K_{**} - K_*^T K^{-1} K_*) \tag{5}$$

Particularly, if y_* is the observation of $f(x_*)$ with noise that is in zero-mean, i.i.d. Gaussian, i.e., $y_* = f(x_*) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$, then we can rewrite the distribution as:

$$\begin{aligned} &y_* | ((x_1, y_1), ..., (x_n, y_n), x_*) \\ &\sim \mathcal{N}(K_*^T (K + \sigma^2 I)^{-1} y, K_{**} + \sigma^2 I - K_*^T (K + \sigma^2 I)^{-1} K^*) \end{aligned} \tag{6}$$

Author Proof

Bayesian Optimization. Bayesian optimization uses Gaussian processes as the approximation of the unknown objective function. Nonetheless, GPs are gradually updated since BO samples the next selected point in each iteration.

Acquisition Functions. Bayesian optimization introduces an acquisition function to decide the next sample point. The simplest way of representing acquisition function $u(\cdot)$ is:

$$u(x) = \mu(x) + m\sigma(x) \tag{7}$$

where m is the weight to balance exploration and exploitation. When m is assigned to a high value, the acquisition function is more aggressive towards where variance is high; on the contrary, the acquisition function will prefer to choose the location where mean is high.

We choose the probability of improvement (PI) with a trade-off parameter ξ as the acquisition function in our work. PI function is defined as:

$$PI(x) = \phi(\frac{\mu(x) - y_{max} - \xi}{\sigma(x)}) \tag{8}$$

The following algorithm is the pseudo code of how we adopted Bayesian Optimization with slight modifications:

Algorithm 1. Bayesian Optimization

1: **if** $t = 1, 2$ **then**
2: Add the upper bound and the lower bound of parameters into GPs as prior knowledge
3: **end if**
4: **for** $t = 3, 4, \cdots, 50$ **do**
5: Optimize the acquisition function: $\mathbf{x}_t = \mathbf{x}_* = \max_{\mathbf{x}} PI(\mathbf{x})$ to find $\mathbf{x}_*$.
6: Sample the objective function: $y_t = y_* = f(\mathbf{x}_*) + \varepsilon$.
7: Expand the data $\mathcal{D}_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, \mathbf{y}_t)\}$ and get GPs updated.
8: **end for**

Random Search. For a better understanding of the problem that we are aiming to optimize we decided to perform a Random Search (RS) with 50 iterations. A full understanding of this 10 dimensional parametric landscape is not a possibility from a real-world perspective, and RS is useful in this aspect. A comparison between RS and BO would be unfruitful, and this is not the purpose of this work. We use function $randi()$ in Matlab and the seed of the random function is set to be associated with the corresponding time, in order to prevent duplicates of generated sets of parameters.

2.2 Experimental Set-Up

The experiment was conducted at a marked arena of 4000×2000 mm (shown in Fig. 4) and we used an oscillator as the controller for the quadruped robot. After

Author Proof

referring to the code in the article [16], we wrote the RS and BO code in Matlab whose version was Matlab 2016b. A DELL OptiPlex 7060 series desktop was used with i7-8700 processor, 12 processor threads and 32 GB internal memory.

Experiments were carried out using RS and Bayesian optimization methods, respectively. We used the evaluation criteria for the vertical distance y (the vertical component of the initial and end positions' displacement) that the robot walked forward within 10 s, and if it went backward, the distance y was negative. A total of 50 iterations were performed for each method. In order to reduce the measurement error, four experiments were implemented for each iteration, and the average of four experiments was taken as the input of each iteration to optimize the parameters.

Oscillators. In the locomotion of animals, the oscillations of the joint angles produced by the muscular activity can have different wave forms which are usually smooth and brutal transitions are uncommon [17]. In order to facilitate practical application, it is necessary to simplify the modeling of locomotion. One way is to approximate the robot's joint angles to sinusoidal variations. According to the gait, these sine waves must have the same frequency but they can differ in phases and amplitude. In other words, a robot's motor activation can be controlled by this simple equation:

$$x(t) = x_0 + Asin(t/\tau + \varphi) \tag{9}$$

where $x(t)$ is the current servomotor state of the robot at time t, x_o is the midpoint of the oscillations for each joint, A is the amplitude of the oscillations, τ is the time constant that determines the oscillation's frequency, and φ is the oscillation phase.

Robotic Platform. The quadruped robot *BayesAnt* mainly consists of two parts and its total size is 350 × 350 × 300 mm (shown in Fig. 1). The body part, printed by onyx and carbon fiber materials, has a protective rope to prevent it from falling during walking. As for the moving parts, an Arduino nano with an Adafruit 16-Channel PWM/Servo Driver is used for controlling eight servomotors, and each leg, made of carbon fiber tubes, has two servomotors. Hip's servomotors are responsible for *BayesAnt*'s movement in the horizontal plane, and knee's servomotors are responsible for its movement in the vertical plane. Therefore, *BayesAnt* can move freely within the 3-dimensional space where its design size is limited.

Based on the real quadrupeds, *BayesAnt*'s four legs were constrained in the whole experiment, e.g., an animal's hip cannot rotate 360°. If quadruped robot *BayesAnt* walk normally, it is obvious that the knee joints should oscillate in a region below the body. On the other hand, each of the four hip joints moves in the same plane. In order to prevent *BayesAnt*'s hips from interfering with each other while walking, each hip accounts for a quarter of a circle (360°), which is 90°.

Author Proof

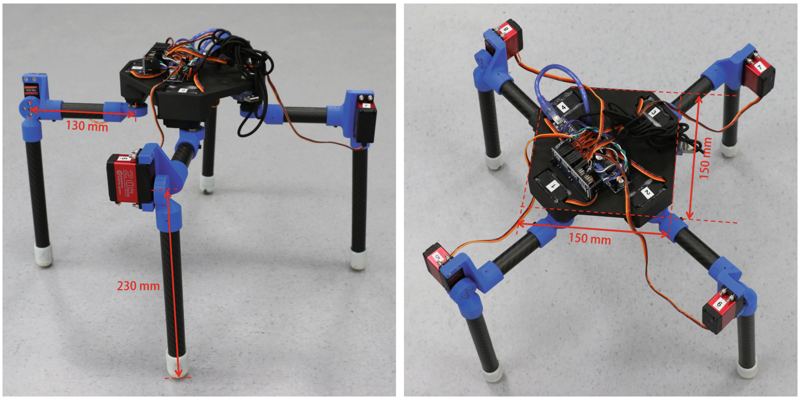

Fig. 1. The general morphology of our quadruped robot *BayesAnt* with a front view on the left and a top view on the right. The left figure shows that the hip limb's length is 130 mm and the knee limb's length is 230 mm. Due to the symmetrical design, only one of their hip limbs and knee limbs is labeled. Similarly, the right figure shows the body's size is 150 × 150 mm, so the overall size is 350 × 350 × 300 mm.

2.3 Parameter Configuration

Our quadruped robot *BayesAnt* has eight servomotors and each servomotor is controlled by an oscillator. Thus, each joint has a sinusoidal equation as shown before. A frequency of 1 Hz ($\tau = 1/(2\pi)$) is chosen as baseline for all joints. This is because it corresponds to the pace of an ordinary animal and it is slow enough for quadruped robot's servomotors to go once 180° back and forth [17]. In that case, it still has three free parameters of the oscillation and in total, there are 24-dimensional parameters.

Considering that each parameter requires a certain range, even for discrete intervals, the search space will be extremely large. For example, if there are four choices for each parameter, then the observation space is about 28 trillion (4^{24}). We decide to do a symmetrical parameter configuration which should be 12-dimensional parameters, but the parameter space is still too large. We turn the four midpoint parameters x_o of four legs' proximal and distal links into two parameters x_i, x_j (see Table 1), reducing our problem to 10-dimensional parameters. Table 1 gives a summary of the corresponding ranges or fixed values that were used. Even so, our search space is still about one million (4^{10}).

For the oscillation central angle, as discussed before, hip's servomotors are responsible for *BayesAnt*'s horizontal movement, and knee's servomotors are responsible for its vertical movement, so we took the midpoint angle of hip and knee as 2-dimensional parameters. For the oscillation amplitude, 4-dimensional parameters are set because of the symmetry of *BayesAnt*'s morphology. And as for the oscillation phase, analogous to the tetrapod's locomotion, 2-dimensional parameters are bilaterally symmetric to the hip and 2-dimensional parameters are diagonally symmetric to the knee.

Author Proof

Table 1. Gait parameters, left(L), right(R), forelimb(F), hindlimb(H).

Parameter		Type	Values[a]	Controls
LF/LH/RH/RF Hip_ang	$x_i(i = 1, 2, 3, 4)$	Free	[255 295 305 345]	Oscillation midpoint
LF/LH/RH/RF Knee_ang	$x_j(j = 5, 6, 7, 8)$		[215 255 295 305]	
LF/RF Hip_amp	$A_i(i = 1, 4)$	Free	[50 55 60 65]	Oscillation amplitude
LH/RH Hip_amp	$A_j(j = 2, 3)$		[60 65 70 75]	
LF/RF Knee_amp	$A_m(m = 5, 8)$		[25 30 35 40]	
LH/RH Knee_amp	$A_n(n = 6, 7)$		[20 25 30 35]	
LF/RF Hip_pha	$\varphi_i(i = 1, 4)$	Free	[−90 0 90 180]	Oscillation phase
LH/RH Hip_pha	$\varphi_j(j = 2, 3)$		[−90 0 90 180]	
LF/RH Knee_pha	$\varphi_m(m = 5, 7)$		[−90 0 90 180]	
LH/RF Knee_pha	$\varphi_n(n = 6, 8)$		[−90 0 90 180]	
Frequency	τ	Fixed	$1/(2\pi)$	Oscillation period

[a] The column described as "Values" is the range of parameters used in our code to control our quadruped robot *BayesAnt*. Since those parameters are used to change the values of Pulse-Width Modulation (PWM) which control the servomotors, they are not the exact angle at which the robot's legs move. For LF knee, LH knee, RH knee, and RF knee, when the PWM value is 300, the part below the knee of *BayesAnt* is 90° to *BayesAnt*'s body. As the PWM values increase, the parts below *BayesAnt*'s knees bend inward. For LF hip, LH hip, RH hip, RF hip, the angles between four legs and body are 45°, while the PWM values are 300. As the PWM values increase, LF hip and LH hip legs move forward. As the PWM values decrease, RH hip and RF hip legs move forward. The *Adafruit_PWMServoDriver* library sets 90 as PWM values to make servomotor turn to 0° and 540 PWM values to make servomotor turn to 270°. We cannot tell the accurate formula for the two parameters, PWM value and the degree change of servomotor, but we can tell that the degree of servomotor increase as the PWM value increase.

3 Results and Discussion

We performed two experiments, one using a RS and the second using BO to select the optimal parameters in 50 iterations. In order to show the parameter changes more clearly, the parameter values are replaced by different colors in Figs. 2 and 3, and the color bar is referred on the right.

We start by exploring the search landscape with RS as we cannot find an analytic solution for real-world 3-dimensional locomotion, so as to have a general understanding of the quality of space combinations. It is important to emphasise that a comparison between BO and RS is not the scope of this paper, as it would be an unfair comparison, and in this sense RS is solely being used to gauge how difficult this problem is. We show the results of 50 random trials in Fig. 2. In

Author Proof

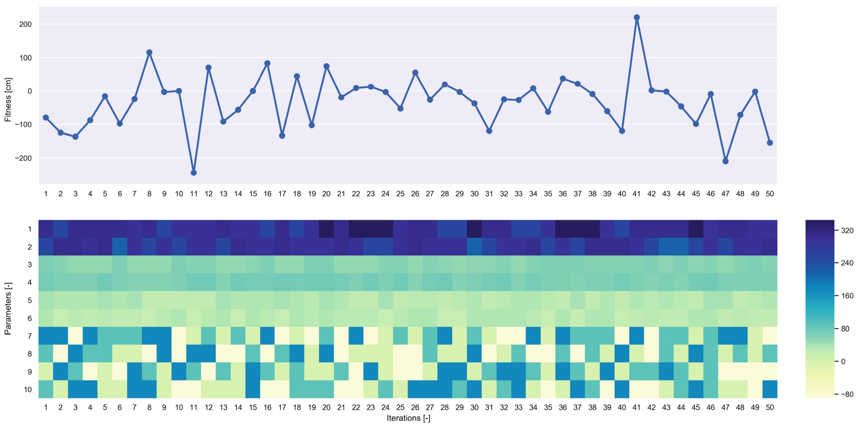

Fig. 2. The upper figure shows that the results of RS oscillate at 0, and negative values mean that our robot moves backward instead of moving forward. The lower figure illustrates how parameters change during each iteration with a heat-map. It can be seen that the color varies widely as every new trial, indicating that RS is mainly an exploratory tool.

only 30% of trials the robot could move forward, in 4% of all trials it passed the one meter mark, and in only one of those trials (trial 42) it could pass the two meter mark. In some trials the robot falls on the floor, where it is assigned zero as their distance. As expected, locomotion is a difficult problem to be optimized, specially from a model-free perspective.

The experimental results of the BO are depicted in Fig. 3, and instead of using random initial parameters we opted for trying the minimum ([255, 215, 50, 60, 25, 20, −90, −90, −90, −90]) and maximum ([345, 305, 65, 75, 40, 35, 180, 180, 180, 180]) parameter combinations, and BO only starts inferring in the third iteration. The reason for this choice is to avoid a "lucky" initial seed bootstrapping good results. Their fitness is −0.04 m and −0.15 m. The iteration 23 represents the optimal parameter combination in 50 iterations, while Bayesian optimization has a maximum distance of 3.61 m at iteration 23 (expressed by speed, it is about 0.36 m/s) with parameter combination [305, 215, 65, 70, 25, 35, −90, 90, 180, −90] and a minimum distance of −1.46 m at iteration 7 with parameter combination [305, 305, 60, 60, 30, 35, 0, 90, 90, 180].

We capture our quadruped robot *BayesAnt*'s gait under the optimal parameter combinations ([305, 215, 65, 70, 25, 35, −90, 0, 90, 180]), found with BO, with a low-speed continuous shooting, to show our experimental results clearer. The walking snapshots (shown in Fig. 4) have shown a forward maximum distance of 3.61 m in 10 s.

As the reader can see from Fig. 3, the gait parameters were initially explored until iteration 7, and then kept the oscillation phase unchanged for the exploitation. After a few trials, it returned to the exploration. Exploration and exploitation are repeatedly cross carried out, and in the end a exploitative behaviour is

Author Proof

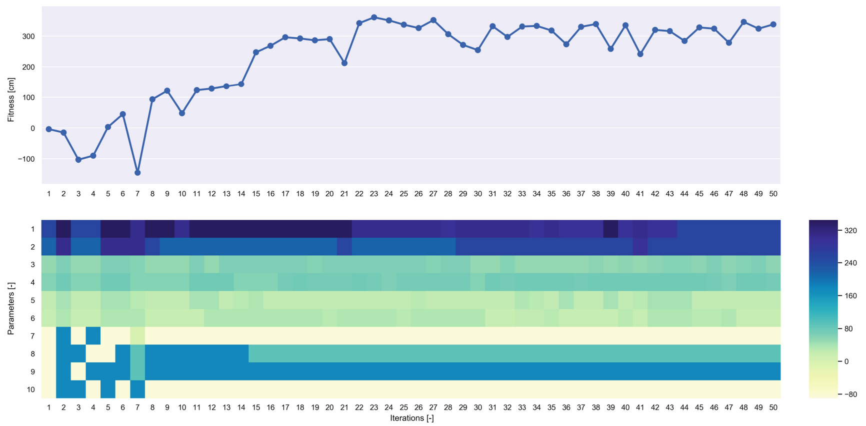

Fig. 3. The upper figure shows the result of Bayesian optimization. It suggests that only 6 trials result in moving backward, and the robot learns to walk further than 2 m after only 15 iterations. Although the method chooses a few missteps (i.e. 7^{th}, 11^{th}, and 21^{st}), we can see that BO gradually improves and stabilizes the behavior. The lower figure illustrates how parameters change during each iteration with heat-map. It can be seen an alternation between exploration and exploitation through the color variation of the parameters after the 7th trial.

dominant due to the Probability of Improvement (PI) acquisition function. It uses a trade-off parameter ξ, so that it starts fairly high early in the optimization to drive exploration, and decreased toward zero as the algorithm continued (exploitation).

A few previous works also used BO on different walking robots, such as the Sony AIBO (Dimensions: $220 \times 130 \times 200$ mm) [9], the bipedal Spring Loaded Inverted Pendulum (SLIP) simulated model [10], the SLIP-based bipedal walker Fox [11], the ATRIAS bipedal robot (Dimensions: 0.9×0.9 m) [12], and the hexapod robot [13]. In [9], they use GPs with the most probable improvement and reach a maximum walking speed about 0.281 m/s. In [10], a bipedal SLIP model and BO are utilized to explore gaits with a step of 0.2 m/s. In [11], they apply GPs with Upper Confidence Bound and reach a maximum walking speed about 0.337 m/s, while ATRIAS obtains a target speed of 0.5 m/s by using BO with Determinants of Gaits (DoG) [12], and the undamaged hexapod robot's maximum speed is 0.32 m/s [13]. As for our quadruped robot *BayesAnt* (Dimensions: $350 \times 350 \times 300$ mm), it reaches a maximum speed of 0.361 m/s. As expected, robots with different dimensions will reach different optimal speeds.

3.1 2D vs 3D Locomotion

SLIP model and ATRIAS both are 2-dimensional locomotion. Especially, ATRIAS moves on a planar around a boom and calculates the Center of Mass (CoM) height at the start and end of each step which avoids the robot falling

Author Proof

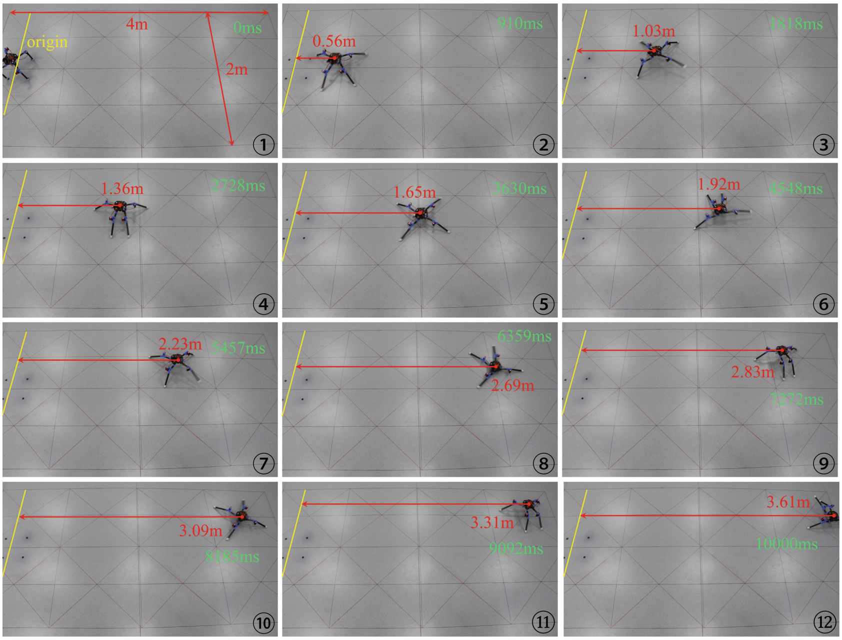

Fig. 4. Snapshots of *BayesAnt* walking on the marked arena within 10 s. We set the origin at the midpoint of the body to facilitate the measurement.

across steps. In our work, *BayesAnt* is open-loop without any sensor and CoM feedback. It is capable of 3-dimensional walking with dynamic locomotion and these properties make our robot easier to fall down than ATRIAS and SLIP. It shows our 10-dimensional parametric search problem during 3-dimensional locomotion is a much tougher problem. Although the hexapod robot is 3-dimensional locomotion, the higher number of legs can guarantee enough foot contact with the floor to preclude the robot from rolling over.

3.2 Prior Models and Kernels

Different robots have different morphologies, and the presence of a computer simulation can accelerate the learning by creating a prior knowledge of how the gait should look like. Different kernel functions can equally affect the convergence speed: experiments with ATRIAS [12] and with the hexapod robot [13] show a faster BO convergence. Both works used simulations to search for thousands of gaits prior to the beginning of experiments, while the work from [12] used a DoG-based kernel with domain prior knowledge. In our experiments we do not use a prior knowledge, and next steps of our research will combine simulation with real-world to assess how much can be improved from this.

Author Proof

3.3 Iterations-Parameters Relationship

The number of iterations required usually increases with dimensions. In [12], the parameters of a 9-dimensional controller on ATRIAS are optimized and the robot successfully walks in 3 trials with 5 runs each trial, yet our quadruped robot reaches a maximum forward distance in 10-dimensional search space at iteration 23. As mentioned previously, A. 2-dimensional problems are easier than 3-dimensional locomotion and B. the presence of a prior knowledge helps the process.

4 Conclusion and Future Work

An automatic parametric search of the gait controller for real-world robot applications is a challenging task. After probing the search space with RS to see how difficult this problem is, we use Bayesian optimization on a quadruped robot which is capable of 3-dimensional locomotion. The results show that there are major advantages of BO, such as data-efficiency (requiring few iterations to reach optimum) and the consideration of noise during the optimization process. Our robot starts from zero knowledge about its morphology and after 23 iterations it reaches optimal behaviour. This methodology, if combined with controllers, can be used to quickly tune parameters after environmental/morphological changes.

Our future work will focus on the adaptation of BO during different degrees of morphological changes. For example, how BO converges after suffering damage, and how to quantify this damage to bootstrap the following learning procedure (as opposed to a new tabula rasa assumption). We are currently working on the usage of BO to investigate the effect of incremental morphological changes on the controller of a quadruped robot [18]. Our current BO model still has lots of room for improvement, such as different kernel functions, acquisition functions and ways to encode the parameter-motor relationship, and we aim to tackle these matters in the near future.

References

1. Poulakakis, I., Smith, J.A., Buehler, M.: Modeling and experiments of untethered quadrupedal running with a bounding gait: the Scout II robot. Int. J. Robot. Res. **24**(4), 239–256 (2005)
2. Seok, S., Wang, A., Chuah, M.Y., Otten, D., Lang, J., Kim, S.: Design principles for highly efficient quadrupeds and implementation on the MIT Cheetah robot. In: 2013 IEEE International Conference on Robotics and Automation, pp. 3307–3312. IEEE (2013)
3. Ijspeert, A.J.: Central pattern generators for locomotion control in animals and robots: a review. Neural Netw. **21**(4), 642–653 (2008)
4. Calandra, R., Seyfarth, A., Peters, J., Deisenroth, M.P.: Bayesian optimization for learning gaits under uncertainty. Ann. Math. Artif. Intell. **76**(1–2), 5–23 (2016)
5. Tedrake, R., Zhang, T.W., Sebastian Seung, H.: Learning to walk in 20 minutes. In: Proceedings of the Fourteenth Yale Workshop on Adaptive and Learning Systems, Beijing, vol. 95585, pp. 1939–1412 (2005)

Author Proof

6. Chernova, S., Veloso, M.: An evolutionary approach to gait learning for four-legged robots. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No. 04CH37566), vol. 3, pp. 2562–2567. IEEE (2004)
7. Seo, K., Hyun, S., Goodman, E.D.: Genetic programming-based automatic gait generation in joint space for a quadruped robot. Adv. Robot. **24**(15), 2199–2214 (2010)
8. Oliveira, M., Santos, C., Costa, L., Ferreira, M.: Quadruped robot locomotion using a global optimization stochastic algorithm, vol. 1389, pp. 500–503 (2011)
9. Lizotte, D.J., Wang, T., Bowling, M.H., Schuurmans, D.: Automatic gait optimization with Gaussian process regression. In: IJCAI, vol. 7, pp. 944–949 (2007)
10. Saar, K.A., Rosendo, A., Llda, F.: Bayesian optimization of gaits on a bipedal slip model. In: 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 1812–1817. IEEE (2017)
11. Calandra, R., Seyfarth, A., Peters, J., Deisenroth, M.P.: An experimental comparison of Bayesian optimization for bipedal locomotion, pp. 1951–1958 (2014)
12. Rai, A., Antonova, R., Song, S., Martin, W., Geyer, H., Atkeson, C.: Bayesian optimization using domain knowledge on the ATRIAS biped. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1771–1778. IEEE (2018)
13. Cully, A., Clune, J., Tarapore, D., Mouret, J.-B.: Robots that can adapt like animals. Nature **521**(7553), 503 (2015)
14. Brochu, E., Cora, V.M., De Freitas, N.: 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)
15. Rasmussen, C.E.: Gaussian processes in machine learning. In: Bousquet, O., von Luxburg, U., Rätsch, G. (eds.) ML -2003. LNCS (LNAI), vol. 3176, pp. 63–71. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28650-9_4
16. Rosendo, A., Von Atzigen, M., Iida, F.: The trade-off between morphology and control in the co-optimized design of robots. PLoS ONE **12**(10), e0186107 (2017)
17. Bourquin, Y., Ijspeert, A.J., Harvey, I.: Self-organization of locomotion in modular robots. Unpublished Diploma Thesis (2004). http://birg.epfl.ch/page53073.html
18. Zhu, J., Li, S., Wang, Z., Rosendo, A.: Influences of incremental mechanical damage on the Bayesian optimization of a quadruped robot. In: 2019 IEEE International Conference on Robotics and Automation (ICRA) Workshop "Towards Real-world Development of Legged Robots". IEEE (2019)