A PID Tuning Method for Hexapod Gait Control Based on Genetic Algorithm

Baisheng Jiang

High School Affiliated to Renmin University Beijing, China

* Corresponding Author Email: alyssa_madrid@elcamino.edu

Abstract. Hexapods are widely used in the field of rescue robots that transport in complex terrains which are difficult for traditional wheeled robots to traverse. Currently, the method for hexapod gait control is often PID, a classical method that is universally exploited in all control systems in general. However, the fine-tuning of PID's hyper-parameters(i.e. kp, ki and kd) mainly rely on chances and the experience of tuners. Therefore, it is both urgent and critical to find an automatic way to get through the tuning process and return optimal results. Aiming at this, this paper puts forward a novel approach to adjust these parameters with genetic algorithm. It first builds a computational model of the hexapod, then the terrain, then designates the input, output and constants of the system, builds the PID closed loop, and finally tunes it using genetic algorithm. The approach described in the paper leads to optimal results in a reasonable amount of time. This makes having hexapods walk on uneven terrain with ease and automacy an alluring possibility.

Keywords: genetic algorithm, PID, hexapod, gait control, complex terrain.

1. Introduction

Multi-legged robots are a research hotspot in today's academic community because of their ability to travel across complex terrain, a function that traditional wheeled robots do not possess. Bipeds, tripods, quadrupeds and hexapods are representative members of the multi-legged robot family. Due to their unique characteristics, gait control is always the key research direction in the field of multi-legged robots.

Since 1997, French scientists have been using genetic algorithm as a tool to generate gait for biped robots. The goal of minimizing energy consumption within each step was achieved [1]. Similarly, in 2004, a group of researchers from Korea generated optimal joint angle trajectories for a biped robot walking a staircase using genetic algorithm. Simulations showed that the biped ascended and descended the stairs stably [2]. Again, in 2008, optimum hip trajectory was successfully generated by a group of Indian researchers for a biped, also using genetic algorithm. The output was a more anthropomorphic and stable robot, and it was proved that the technique can be utilized for generating real time trajectory [3]. Recently, in 2017, a group of researchers from China made comprehensive optimization of wall-climbing possible by genetic algorithm for wall-climbing bipeds [4]. And in 2019, researchers from China used the same genetic algorithm to plan the movement of squat down so as to make it stable and gliding [5].

The research on genetic algorithm on hexapods generally started later than that on bipeds. It was started by authors from Serbia and Hungary, in 2010, optimizing hexapod walking by genetic algorithm, and authors marked this work as a starting point for future research [6]. Then in 2011, a group of Estonian scientists used a genetic-gravitational hybrid method to generate hexapod walking gait, and better results were achieved [7]. Meanwhile, in 2014, some scientists from the USA achieved self-sufficiency on a hexapod using a cyclic genetic algorithm, i.e., telling it the correct time to recharge itself by the algorithm. Although the study has little to do with gait control, the success on self-sufficiency contributed greatly to the overall accomplishment in hexapod robots' application [8]. Again, in 2015, a group of researchers from China used genetic algorithm for parameter tuning of CPGs (Central Pattern Generators) for hexapod gaits. The CPG model used in the study was based on Van Der Pol Equation [9]. Then in 2016, Indian scientists made it possible to generate gait for hexapods with one leg damaged. The damaged hexapod performed close to undamaged robots [10].

Finally, in 2020, a group of scientists from Rochester Institute of Technology made their hexapod learn to generate its own gait when it detected N motors in itself were inoperable [11].

This research gives today's academic field great insight and past experience. Among the whole family of multi-legged robots, hexapods are the most symmetrical, simple in structure, generally small in size, and easier to maintain its own balance than bipeds or quadrupeds. Therefore, the author considers it most worth researching, especially when put into use as rescue robots in situations dangerous for humans to go in first. It can be built to search for both lives and potential danger sources in fire ruins and earthquake ruins. Because it has the ability to travel across uneven terrain, it has the potential to perform better than traditional wheeled robots.

However, the past research on genetic algorithm for hexapod gait control all have one universal defect: they are too slow to be used to control the hexapod's gait online. Therefore, in this paper, the author puts forward a novel approach: using genetic algorithm to pre-tune the kp, ki and kd parameters of the robot, and later controlling the robot's gait (i.e., the third joint angle of each leg on the robot) with these parameters real-time.

The goal of this research is to find the optimal method to determine the kp, ki and kd parameters for the hexapod's gait control in uneven terrain, in which "optimal" includes both speed and accuracy.

In the following parts of this paper, the methods (including mechanical structure, PID gait control function and parameter tuning with genetic algorithm), the results (final converged values of PID parameters and simulated hexapod gait), the discussions and the conclusion will be provided with full detail.

2. Methods

2.1. Hexapod

The hexapod used in this research consists of 6 identical legs and 2 boards serving as the main body (Figure 1, left). It has a total mass of 1.844kg. The Coxa, Femur and Tibia of each leg are 32.0mm, 54.51mm and 92.91mm respectively (Figure 1, right). The angle limitations of the three joints are (-45, 45), (-45, 75) and (-60, 60) degrees respectively.

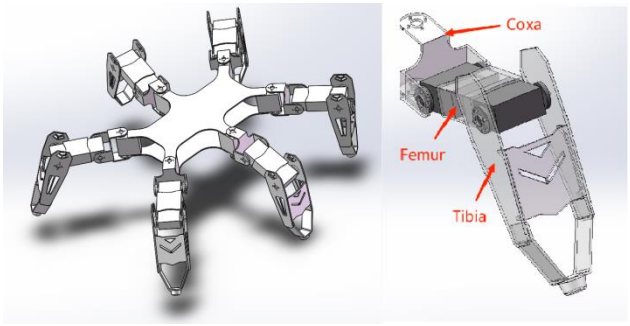

Fig. 1 Structure of the Hexapod (Picture credit: Original).

2.2. Simulation Environment

The simulation environment chosen for this work is Python 3.8. The Animation and Pyplot modules of Matplotlib is used to plot the movements of the body and legs of the hexapod. A high-order 3D curve function F(x, y) is used to simulate an uneven surface on which the hexapod walks. The simulation environment looks like the following (see Figure 2).

$$F(x,y)=3(1-x)^2e^{-(x^2)-(y+1)^2}-10(x/5-x^3-y^5)e^{-x^2-y^2}-\quad (1/3)^{e^{(-(x+1)^2-y^2)}} \tag{1}$$

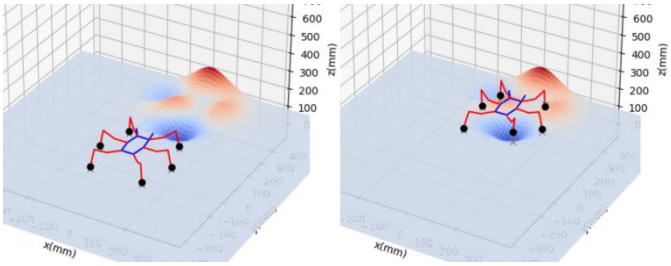

Fig. 2 Simulation of Hexapod Walking on Uneven Terrain (Picture credit: Original).

The blue lines illustrate the body of the hexapod, and the red lines illustrate the legs of the hexapod. The black dots indicate the positions of the tips of each leg. The gray crosses indicate the positions where the line passing through the black dots parallel to the z-axis intersects with the uneven surface. The goal of the simulation is to make the black dots and their corresponding gray crosses as close to each other as possible, which means that the hexapod is able to walk as smoothly as possible, with each tip touching the ground and the body remaining level to the flat x-y surface.

2.3. PID Gait Control

A table is first generated as a basis to control the hexapod's movements when it is moving forward, backward, left, right or turning left or right on a flat surface. Within each time point in each step, each of the 18 (3 joints times 6 legs) joints' angles are predefined. There are also functions performing as interpolation processes that divide each interval between two time points into smaller intervals and calculate the respective joint angles for each joint to make the simulation operate more smoothly.

Then, the PID algorithm is put forward for gait control adaptation on uneven terrains.

$$u = kp*e + ki*(\int e\,dt) + kd*(de/dt) \tag{2}$$

The standard PID algorithm divides the input into three parts: a multiple of the error, a multiple of the integral (i.e. accumulation) of the error, and a multiple of the derivative (i.e. rate of variation) of the error. In cases where the variation of the error is too unpredictable (like in this particular case), the kd part should be set to zero for a better result.

This paper assumes that there is a pressure sensor at each tip of the leg. If the hexapod's mass is evenly distributed among each leg, the pressure function on each leg will be close to zero. If the pressure on one leg is much higher than a normal value, the PID algorithm automatically adjusts the pressure by raising the Tibia a bit. On the contrary, if the pressure on one leg is much lower than a normal value, the PID algorithm automatically adjusts the pressure by lowering the Tibia a bit.

Several different pressure functions had been tested to gain the best results before deciding on the final pressure function. The first pressure function that was tested was a simple linear function:

$$p = h - z \tag{3}$$

Where p stands for pressure, and h stands for ground height (derived from F (x, y)), while z stands for absolute height of the foot(derived from given joint angles and forward kinematics).

This simple linear function did not work very well. Specifically, PID parameters derived from the genetic algorithm introduced in the following chapter, though showing good convergence, resulted in neither normal gait nor adherence of foot tips to the complex ground.

Therefore, the pressure function was then changed into a cubic curve function:

$$p = (h - z)^3 \tag{4}$$

The reason a cubic curve function was introduced here was that it is noticed using the linear function above, the model was not sensitive enough to distance between the foot tips and the ground, resulting in very large distance between them now and then when walking. However, when the cubic curve was applied, another undesired thing happened: the loss function very often ended up in rocketing, and it was rather hard to come to convergence.

It was clear that something was needed to come in between. However, a simple hyperbolic function would be very different from either a linear function or a cubic function, as it eliminates the sign of (h - z), which was actually very important as it shows the direction of the desired compensation.

Therefore, combining the advantages of the hyperbolic function and the cubic function, the final pressure function is as follows:

$$p = (h - z)^2 \cdot \text{sgn}(h - z) \tag{5}$$

Where sgn(x) takes the sign of x, giving 1 when x is positive and -1 when x is negative, specifically, 0 when x is zero. This pressure function is proved to be the best function tested in the past experiments.

This pressure function on each leg is the only input and the joint angle of the third joint(i.e. the joint between the Femur and the Tibia) is the output. The angles of the other two joints on each leg are not modified by the algorithm, and they stick to the predefined trajectories. In this experiment, for each step the hexapod moves forward, an algorithm is applied to each leg(assuming the pressure function of each leg is independent from each other): for each epoch in 50 epochs, calculate the current pressure function of that leg, adjust the joint angle of the third joint with the PID function as in (1), and then re-calculate the pressure function of that leg in the next epoch. After 50 epochs, the joint angle of the third joint converges to a reasonable value.

$$\text{third\_angle} \mathrel{+}= \text{kp}*\text{pressure} + \text{ki}*\text{pressure\_int} \tag{6}$$

In (1), pressure_int means the integral of the pressure function.

2.4. PID Tuning with Genetic Algorithm

This experiment used genetic algorithm to find the best kp and ki. The tuning process is done before the simulation, i.e., the testing process.

Genetic algorithm, or GA, is a modern method for determining parameters in an equation through a machine learning process. The process includes generation of a random “population”, and iterations through the following steps in each epoch: 1) crossover(exchanging bits), at a certain rate, between “parents” selected from the population; 2) mutation(randomly flipping bits), at a usually lower rate, from the derived “child”; 3) computation of the fitness function for the current derived “DNA”, which can be translated into a meaningful pair of parameters; 4) selection for the optimal child population of this epoch, in which “optimal” means derivation of the maximum or minimum fitness function accordingly. After a certain time of iteration, the population converges to one that can be translated into an optimal pair of parameters, i.e. the desired result.

For the last step, the selection, there is an equation determining the probability each individual DNA from the population is to be selected:

$$P(i) = \text{fitness}(i)/\sum \text{fitness}(k) \tag{7}$$

This experiment used a DNA size of 24 and a population size of 200. It used a crossover rate of 0.8 and a mutation rate of 0(as zero mutation is enough to find an optimal value and makes the algorithm faster to converge). It went through 50 generations. The boundaries of kp were [0, 0.01], and the boundaries of ki was [0, 0.001], given by experience.

In the middle of the training process (i.e., epoch 21/50), the 3D visualization showed something like this (see Figure 3):

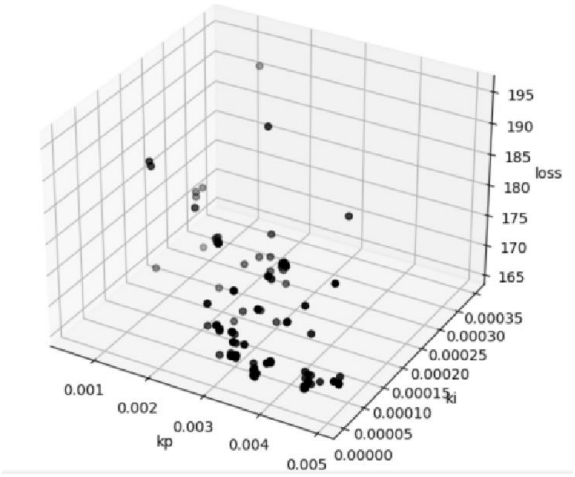

Fig. 3. Visualization of PI Parameter Tuning Using Genetic Algorithm (Picture credit: Original).

It can be seen that kp is converging to a point between 0.004 and 0.005, and ki is converging to a point very close to 0.

3. Results

The experiment described as above resulted in a kp of 0.00463255 (between 0.004 and 0.005) and a ki of 1.16080053e-06 (very close to 0). While the population size and number of generations were altered, the range of the results remained roughly unchanged.

To compare the genetic algorithm with classical methods, the author of the paper also used Newton's method to check out the converged values of kp and ki. The results are considered to be within the reasonable range. By Newton's method, the final result of kp is 0.00502315 (only slightly above 0.005), while the value of ki is set to 0 by hand (see Table 1).

Table 1. Results Generated from Newton's Method and Genetic Algorithm are Close to Each Other

Method Used/Final Results	kp	ki
Newton's Method	0.00502315	0 (set by hand)
Genetic Algorithm	0.00463255	1.16080053e-06

While the numerical results of genetic algorithm and Newton's method seem consistent to each other, they do not generate the best results for the hexapod walking on uneven terrain, at least apparently. The gait in Figure 2 is derived by a pair of hand-tuned parameters: kp = 0.001, and ki = 0. Instead, the pair of parameters generated by either genetic algorithm or Newton's method give out rather abnormal gaits (Figure 4):

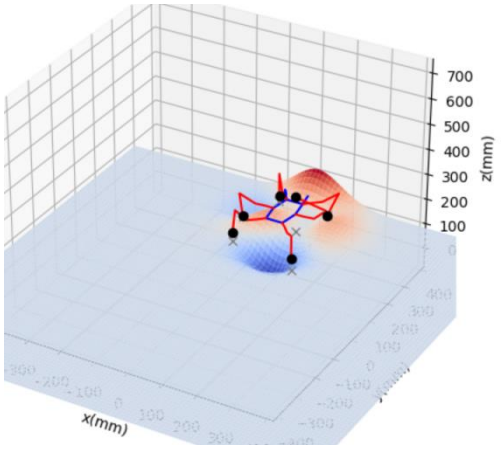

Fig. 4. Hexapod Showing Abnormal Gait in Simulation (Picture credit: Original).

Not only the look of the gait is very strange, but also the black dots (tips of the legs) do not match the gray crosses (the intersection of the vertical line passing through the leg tips and the surface) well.

4. Discussion

This work sets a novel approach to combine the traditional PID controlling method with the advanced genetic algorithm, using the latter to fine-tune parameters of the former to achieve optimal results. The advantage of this method is that it is quick, do not require heavy computation, can be run on a personal computer, and can fit in the uneven terrain the hexapod walks on in real-time. However, there are quite unexpected outcomes, and some analysis can be given according to the author's past experience.

The fact that genetic algorithm converges to a point consistent with the Newton's method indicates that this is still a viable method to find optimal kp and ki values. However, the poor performance of this pair of parameters indicate that a better loss function may be needed.

The work also did not involve fine tuning of the first two angles, due to lack of experience to tune multiple parameters with PID at the same time.

Moreover, the body of the hexapod is kept level all the time as a requirement. While this sounds reasonable (as sometimes the cargo carried by the hexapod needs a level ground so as not to spill out), it greatly limits the ability of the hexapod to travel across terrains with higher variations (eg. to climb a steep slope).

5. Conclusion

This research dips into the possibility of pre-tuning PID parameters with genetic algorithm for a hexapod walking on an uneven terrain. It has shown that the results can converge, and they do converge to a point where the classical Newton method also converges into, indicating this work is on the right path. However, the absence of a good loss function makes the final presentation rather awkward, indicating that there is still a long way to go for future work.

There are several directions where future work may aim at:

Finding a better loss function for genetic algorithm.

Trying to involve tuning of the first two joint angles, to create more flexibility, instead of only the third joint angle.

Eliminating the need to keep the main body part level all the time, allowing it to tilt to create more flexibility while the hexapod is walking on uneven terrain.

If work is further done in these three aspects, there is chance that automated hexapods walking on uneven terrain will evolve into a more mature state.

References

[1] G. Cabodevila and G. Abba, "Quasi optimal gait for a biped robot using genetic algorithm," 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, Orlando, FL, USA, vol.4, pp. 3960-3965, 1997.

[2] K. S. Jeon, O. Kwon and J. H. Park, "Optimal trajectory generation for a biped robot walking a staircase based on genetic algorithms," 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Sendai, Japan, vol.3, pp. 2837-2842, 2004.

[3] A. D. Udai, "Optimum Hip Trajectory Generation of a Biped Robot during Single Support Phase Using Genetic Algorithm," 2008 First International Conference on Emerging Trends in Engineering and Technology, Nagpur, India, pp. 739-744, 2008.

[4] Q. Zhan, L. Yang, Y. Zhang, Y. Ma and A. Rahmani, "Research on Torque Optimization of Biped Wall Climbing Robot Based on Genetic Algorithm," 2017 International Conference on Computer Technology, Electronics and Communication (ICCTEC), Dalian, China, pp. 1030-1035, 2017.

[5] J. Zhang, X. Luo, Z. Xie, L. Li and Y. He, "Research on Motion Planning of the Biped Robot based on Genetic Algorithm," 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, pp. 944-949, 2019.

[6] Z. Pap, I. Kecskés, E. Burkus, F. Bazsó and P. Odry, "Optimization of the hexapod robot walking by genetic algorithm," IEEE 8th International Symposium on Intelligent Systems and Informatics, Subotica, Serbia, pp. 121-126, 2010.

[7] F. Seljanko, "Hexapod walking robot gait generation using genetic-gravitational hybrid algorithm," 2011 15th International Conference on Advanced Robotics (ICAR), Tallinn, Estonia, pp. 253-258, 2011.

[8] G. Parker and R. Zbeda, "Learning Area Coverage for a Self-Sufficient Hexapod Robot Using a Cyclic Genetic Algorithm," in IEEE Systems Journal, vol. 8, no. 3, pp. 778-790, Sept. 2014.

[9] W. Li, W. Chen, X. Wu and J. Wang, "Parameter tuning of CPGs for hexapod gaits based on Genetic Algorithm," 2015 IEEE 10th Conference on Industrial Electronics and Applications (ICIEA), Auckland, New Zealand, pp. 45-50, 2015.

[10] A. Manglik, K. Gupta and S. Bhanot, "Adaptive gait generation for hexapod robot using Genetic Algorithm," 2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), Delhi, India, pp. 1-6, 2016.

[11] J. Kon and F. Sahin, "Gait Generation for Damaged Hexapods using a Genetic Algorithm," 2020 IEEE 15th International Conference of System of Systems Engineering (SoSE), Budapest, Hungary, pp. 451-456, 2020.