EVOLVING CONTROL PROGRAMS FOR A BIPED STATIC WALKER

Chanchai CHAISUKKOSOL, Prabhas CHONGSTITVATANA
Department of Computer Engineering
Chulalongkorn University, Bangkok 10330. Thailand
Tel: (662) 218-6983 Fax: (662) 218-6955
42702899@student.netserv.chula.ac.th, prabhas@chula.ac.th

***Abstract**: This work presents an automatic method to synthesis robot control programs. The proposed method is based on Evolutionary Computation. The problem of biped robot walking is chosen to test the proposed method. Walking motion is divided into six stages and the evolution of control programs is carried out stage-by-stage. The locomotion is restricted to forward walking on the flat and smooth surface with static balance. The synthesis process consists of both simulation and the experiment with a real robot. The result of the experiment shows that the biped walking is achievable and stable.*

***Keywords:** Evolutionary computation, biped robot walking, static balance walking.*

1. INTRODUCTION

Automatic programming for a robot to achieve a task has been a long-term goal of robotic research community. Programming a robot by human is difficult and error prone. Modern robots are very complex, some robot has sophisticated mechanisms that enable it to perform human tasks such as the humanoid robot P2 by Honda (Hirai, et al, 1998). The limitation of using these robots is the difficulty in programming them to achieve a desired task.

Evolutionary Computation is a family of algorithms, some of which can produce solutions in the form of "programs". It is applicable to robot problems. Many works have been demonstrated, for example, (Koza and Rice, 1992; Davidor, 1990; Chongstitvatana and Polvichai, 1996). Evolutionary Computation can be regarded as a weak search method. It is effective for a wide range of problems such as symbolic regression, job scheduling, robot control and so on.

Evolutionary Computation is a search method based on population. A number of candidate solutions are evolved generation by generation to converge to a final solution. The search is guided by the measure of goodness of candidate solutions, called "fitness function" which is defined for a particular problem to be solved.

Many problems in robot program synthesis that have been attempted using evolutionary computation are the problems that have low number of degree of freedom and mostly are the work in simulation. This is because of the high cost of computation and the huge number of candidate solutions to be evaluated. It is well known that transferring the solution from simulation to the real world is not very successful (Brooks, 1991a). Many aspects of the real world can not be sufficiently simulated. To improve the success rate, the real world should be engineered such that the simulation can predict the effect in the real world with high degree of accuracy. This is difficult if not impossible in many tasks which robots are intended to be used.

This work proposes to synthesis programs for a biped static walker. This task is chosen because it contains high degree of freedom. A walking robot is interesting because it can travel in many terrain that are not accessible to a typical wheel-based mobile robot. A biped robot is also more appropriate in the area that is constructed for human, such as in a car, in a tunnel, on an elevator etc. A biped is deemed to be more difficult to control than a multilegged robot as it has to perform balancing with minimum degree of redundancy. Genetic Algorithm is used to synthesis programs. The walking task is divided into stages and the program is synthesized stage-by-stage. In each stage, the solutions from simulation are validated using the experiment in the real world. These validated solutions become the initial state of the next stage of the synthesis. This is the key to improve the transferring of solutions form simulation to the real world. The subsequent sections explain the proposed method in more details.

2. RELATED WORKS

There are many works on generating robot programs. Genetic Algorithms (GA) and Genetic Programming (GP) two of the most popular methods in Evolutionary Computation are widely used. Hirai, et al, (1998) developed a humanoid robot that has full body, head arms, and legs. It could walk perfectly like human, it could walk up and down the staircase, turn left and right, and walk on any surface. This robot, however, used manual programming. Chongstitvatana and Polvichai (1996) demonstrated the automatic generating of robot programs by using Genetic Programming (GP). The robot is 3-joint arm moving in two dimensions. Experiments were performed in simulation, and the results were validated in a real robot.

There are many works on generating robot program to control biped locomotion. Zheng, et al, (1988) developed biped walking from a level surface to sloping surfaces with positive gradients. Inaba, et al, (1995) constructed an ape-like biped that can walk with static balance. Kun and Miller (1997) applied neural network to perform adaptive static balance of biped walking.

Regarding works that apply GA to solve the biped walking problem. Cheng and Lin (1995) developed a walking robot with dynamic balance. In his work, GA is applied to search for control gains and nominal trajectory for a 5-link biped locomotion. The aim is to walk in different constraints, such as, walking on an incline surface, walking at a high speed, and walking with a specified step size. The biped is experimented in simulation. Rodrigues, et al, (1996) used GA to find the minimum torque that is necessary for walking. The fitness function is defined as similarity between the ideal posture and the actual posture. The experiment is also performed in simulation. Arakawa and Fukuda (1996, 1997) focused on using GA to produce a natural motion trajectory and optimize walking energy. The learning system was performed in simulation and the result was confirmed by the real robot.

Most researches experimented only in the simulation. In this work, there are both simulation and real world included in the experiment.

3. GENETIC ALGORITHMS

Genetic Algorithms (GA) (Goldberg, 1989) is a subfield of Evolutionary Computation. A candidate solution, called “individual”, is a string. The process of GA is shown in Fig 1.

```
Initial Population
Repeat
    Fitness Evaluation
    Selection
    Crossover
    Mutation
until (Terminated condition)
```

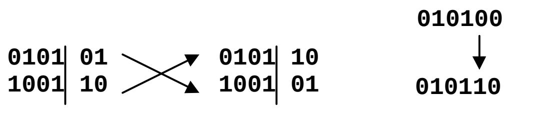
Fig. 1. Process of GA.

First, a number of individuals, called “population”, are initialized as the first generation. All individuals are evaluated to find their fitness. The genetic operators—crossover, and mutation—are applied to produce the next generation. The population in the next generation is evaluated again. Selection, crossover, and mutation are executed repeatedly until the terminated condition is achieved.

There are many methods to select individuals. Generally, individuals are selected according to fitness value. That means the higher fitness individual is more likely to be selected than the lower one. Nevertheless, this method can cause all individuals to become similar to each other. In this work, we use a selection, called “Combined Rank Selection” (Winston, 1992).

The combined rank selection works as follows. The individuals are ranked by their fitness. The highest fit individual is then select as the first candidate. The Hamming distance of all other individuals from the selected candidate are then calculated. The individuals are then ranked by their distance measures. The two ranks--fitness and distance--are added then the next candidate is selected based on this combined rank. This process is repeated for the desired number of candidates. Using this selection, the diversity of the population can be maintained.

Crossover is an operation that exchanges some parts of an individual with another one (see Fig. 2.a). For mutation, some bits of a string are randomly changed (Fig. 2.b). The mutation occurs by a small probability, for example 0.001.



(a) Crossover (b) Mutation

Fig. 2. Genetic operations

4. EXPERIMENT

The objective is to synthesis the biped robot control program automatically. This work restricts the walking task to the biped that can walk forward on the flat and smooth surface with a static balance.

The experimental biped is 25 centimeter high, and the area of the sole is 4.5×5.0 cm^2. It has two hips, two knees, and two ankles, rotated in sagittal plane (Fig. 3). The biped does not have a torso, but it has a tail moving in frontal plane. The reason for using a tail instead of a torso is that the tail will lower the biped’s center of gravity (C.G.), so the biped can keep its balance easier.

An individual contains two fields: a length, and a sequence of walking commands.

Length	a sequence of walking commands

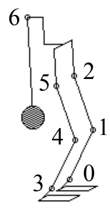


Fig. 3. Biped construction

The sequence of walking commands has the form:

m: r

where ‘m’ is motor command {0+, 0-, 1+, 1-, …, 6+, 6-}. The biped has 7 motors numbered 0-6. The signs ‘+’ and ‘-’ mean increasing or decreasing angle of the motor by ‘r’, $0 \le r \le 150$.

Walking motion of one step is divided in to six stages (Fig. 4). GA is used to synthesize control program for each stage step-by-step, called “stage evolution” (Brooks, 1991b). With this approach, the fitness function can be set appropriately with the subgoal of each stage. Thus, the final solution can be achieved more rapidly.

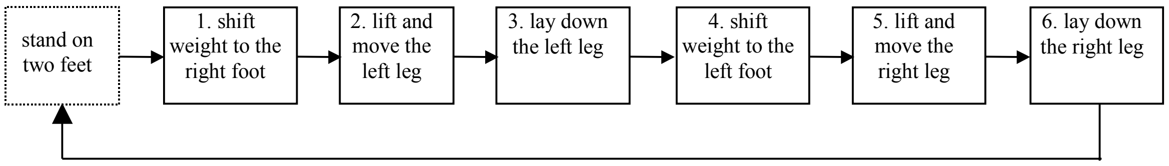

stand on two feet
1. shift weight to the right foot
2. lift and move the left leg
3. lay down the left leg
4. shift weight to the left foot
5. lift and move the right leg
6. lay down the right leg

Fig. 4. Six stages of walking motion.

The initial biped posture is standing on two feet. In the first stage, the robot shifts its weight to the right foot. In the second stage, it lifts the left leg. The third stage, it lays down the left leg. The fourth stage, the robot shifts its weight to the left foot. The fifth and sixth stage it lifts the right leg and lays it down. After the final stage, the posture is adjusted to be similar to the initial posture. The sequence of control can be repeated to create a continuous walk.

There are two types of fitness function: general fitness function, and particular fitness function. Both fitness functions are minimized function. The general fitness function consists of three variables:

$$Fit = \frac{k_1F + k_2R}{k_3\,T}$$

where $F = 1$ when the robot falls otherwise 0, $R = 1$ when the robot turns otherwise 0, T is the duration that the robot can achieve stable walk, k_1, k_2, k_3 are appropriate constants. The general fitness function promotes the behavior that is stable and walk straight without turning.

The particular fitness function for each stage is shown in Table. 1.

$F_1 = \Delta S_R$	$\Delta S_R = \sqrt{(cg_x - pr_x)^2 + (cg_z - pr_z)^2}$
$F_2 = k\Delta z + \Delta y$	$\Delta z = \lvert pl_z - (pr_z + step\_size)\rvert$ $\Delta y = \lvert pl_y - ground\rvert$
$F_3 = \Delta y + penalty$	$\Delta y = \lvert pl_y - ground\rvert$
$F_4 = \Delta S_L$	$\Delta S_L = \sqrt{(cg_x - pl_x)^2 + (cg_z - pl_z)^2}$
$F_5 = k\Delta z + \Delta y$	$\Delta z = \lvert pl_z - pr_z\rvert$ $\Delta y = \lvert pr_y - ground\rvert$
$F_6 = k\Delta y + \Delta z$	$\Delta z = \lvert pl_z - pr_z\rvert$ $\Delta y = \lvert pr_y - ground\rvert$

Table. 1. Particular fitness function for each stage

where

cg_x, cg_z	is the position of C.G. by X and Z axis
pl_x, pl_y, pl_z	is the position of center of the left sole by X, Y, and Z axis
pr_x, pr_y, pr_z	is the position of center of the right sole by X, Y, and Z axis
step_size	is the length of stride (2.5 cm. in the experiment)
penalty	is the penalty value if the robot shifts its weight from the right foot.
ground	is the position of ground level (Y axis)
k	is constant

The motivation for each particular fitness function is as follows. For the first stage, the robot must shifts its C.G. to the right foot. The function F_1 measures the distance between C.G. and the center of the right foot (ΔS_R). In the second stage, the robot lifts the left leg and moves it forward. The function F_2 measures the distance of the left foot in front of the right foot (Δz). The variable Δy controls the height of the left foot from the ground. The *step_size* is used to limit the length of stride to prevent the subsequent difficulty in transferring the weight to the right foot in the fourth stage. In the third stage, the left foot is laid down to the ground. The function F_3 measures the height of the left foot from the ground (Δy). The *penalty* value is used to prevent the robot from shifting its weight to the left foot. If this happens the robot will sway its body to the left side. In the fourth stage, the robot moves its C.G. from the right foot to the left foot. F_4 is similar to F_1 but alternate left and right. The fifth stage is similar to the second stage but alternate left and right. The right foot is not placed forward, the *step_size* is zero. The robot lays down the right foot in the sixth stage. The variable Δz is used to prevent the right leg to move backward or forward.

Evolving the control programs in the simulation is necessary. The experiment with an actual robot takes a very long time as the number of candidates to be evaluated is up to 100,000. However, the solution from simulation alone does not yield programs that works in the real world. The experiment with the real robot is combined in the simulation to increase the success rate. At the end of each stage of evolution, the experiment with the real robot is performed to validate the solutions.

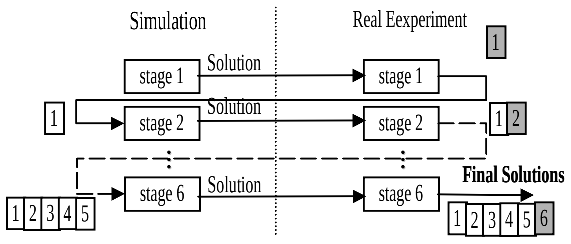


Fig. 5. Simulation + real world

GA is run in each stage of evolution in the simulation (Fig. 5). GA generates the solutions with some variations. As the simulation ignores many aspects of the real world, many solutions from the simulation simply fail. However, some solution has a chance of success. The experiment with the real robot is performed to select only the solutions that work in the real world to be further evolved to the next stage. Each validated result becomes an initial state of the next walking stage. After six stages, the complete solutions will emerge. The number of different solutions from each stage is retained to the next stage hence there are many different complete solutions at the end. This method combines the advantage of simulation (speed) with the advantage of the experiment with the real robot (validity).

For the experiment with the real robot, the human observation is used to score the behavior of the robot. There are 2 types of criteria in observing the real behavior: *general criteria* used in every stages, and *particular criteria* used in each stage. The general criteria judges the stability and the direction of the walk. The observer asks the questions "Does the robot fall?" and "Does the robot turn?". The particular criteria are set differently for each subgoal of each stage, as shown in Table 2.

stage	**particular criteria**
1	Is C.G. shifted to the right foot?
2	Is the left leg lift forward?
3	Is the left foot on the ground?
4	Is C.G. shifted to the left foot?
5	Is the right leg move forward?
6	Is the right foot on the ground?

Table 2. Particular criteria for each stage

The GA parameters are shown in Table 3.

Population size	500
Generation	200
Crossover probability	1.0
Mutation probability	0.001

Table. 3. GA parameters

Because an individual can contain redundant motions, such as moving a joint back and forth or repeating the same joint motion, edit operations are performed after its fitness evaluation. The edit operations are 1) eliminate redundant motions and 2) simplify repeating joint motions. These operations help to maintain the compactness of the representation.

5. RESULT

The robot can walk continuously more than 15 steps, with the speed 40 second per step. Figure 6 shows an example of a full step. Figure 7 shows the movement of C.G. during a one-step walk. The solid line shows the foot that is on the ground. The dotted line shows the foot that is lifted. This figure is drawn from the simulation result. It can be seen that the stability of biped locomotion is marginal, especially in the stage 2 - 4.

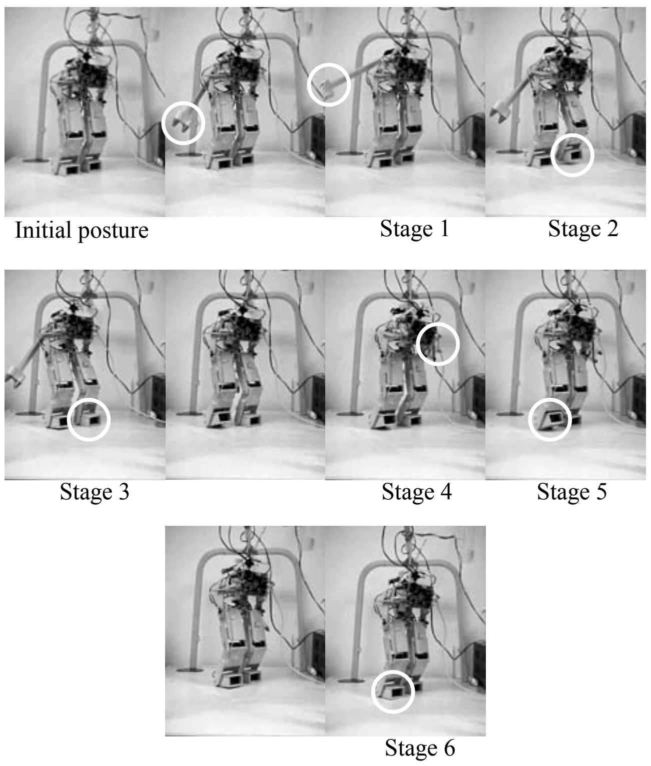


Fig. 6. Result.

At the end of each stage in the simulation, 20 individuals are selected to be validated with the real robot. An average number of successful individual in the experiment with the real robot of each stage is 7. We found that even without the general fitness function, the final solution could still be achieved.

The fourth stage is the most difficult stage to evolve. The robot must transfer its weight to another foot. It becomes more difficult when the length of stride is large. The length of stride is determined by the fitness function in the second stage. Sometimes, the unexpected behavior emerges in the fourth stage such as moving the leg backward before the weight transfer.

6. CONCLUSION

In this work, we investigate a method to automatically generate control programs for a walking biped. Walking motion is divided into six stages. GA is used to synthesize the robot control program stage-by-stage. The fitness function is set differently and appropriately in each stage. This work uses simulation combined with the experiment in a real robot. The results show that the real robot can achieve a stable and continuous walk.

The experiment in the real world is used to select and validate the result from the simulation. The cooperation between simulation and real world experiment is the key to achieve a solution that works in the real world.

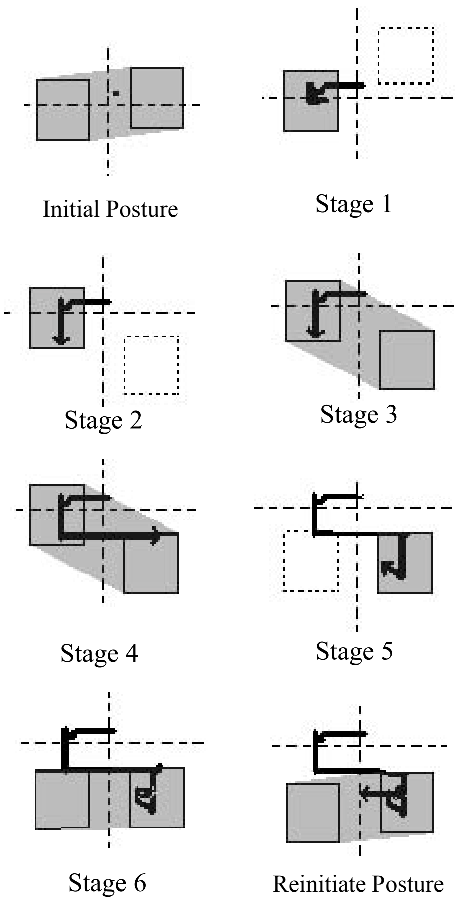


Fig. 7. Movement of C.G.

7. REFERENCES

Arakawa T., Fukuda T. (1996). “Natural motion trajectory generation of biped locomotion robot using genetic algorithm through energy optimization”, Proc. of IEEE Int. Conf. on Systems, Man and Cybernetics, Vol. 2, pp. 1495 -1500.

Arakawa T., Fukuda T. (1997). “Natural motion generation of biped locomotion robot using hierarchical trajectory generation method consisting of GA, EP layers”, Proc. of IEEE Int. Conf. on Robotics and Automation, Vol. 1, pp. 211 - 216.

Brooks R. A. (1991a). “Artificial Life and Real Robots”, Towards a Practice of Autonomous Systems: European Conference on Artificial Life, Paris, France, MIT Press, December, pp. 3 - 10.

Brooks R. A. (1991b). “Intelligence without representation”, Artificial Intelligence, 47, pp. 139 - 160.

Cheng M.-Y., Lin C.-S. (1995). “Genetic algorithm for control design of biped locomotion”, Proc. of Int. Conf. on Intelligent Systems for the 21st Century, Vol. 2, pp. 1315 -1320.

Chongstitvatana, P., Polvichai, J. (1996). “Learning a visual task by Genetic Programming”, Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Vol. 2, pp 534 - 540.

Davidor Y. (1990). “Robot programming with a genetic algorithm”, Proc. of IEEE Int. Conf. on Computer Systems and Software Engineering, pp. 186 - 191.

Goldberg D. E. (1989). “Genetic Algorithm in Search, Optimization, and Machine Learning”, Addison Wesley.

Hirai K., Hirose M., Haikawa Y., Takenaka T., (1998). “The development of Honda humanoid robot”, Proc. of IEEE Int. Conf. on Robotics and Automation, Vol. 2, pp. 1321 - 1326.

Inaba M., Kanehiro F., Kagami S., Inoue H., (1995). “Two-Armed Bipedal Robot that can Walk, Roll Over and Stand up”, Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Vol. 3, pp. 297 - 302.

Koza J. R., Rice J. P. (1992). “Automatic programming of Robots using Genetic Programming”, in Proc. 10th National Conf. on Artificial Intelligence, pp. 194 - 201.

Kun A. L., Miller W. T. (1997), “Adaptive Static Balance of a Biped Robot Using Neural Networks", Proc. of the Int. Conf. on Robotics and Manufacturing (IASTED), Cancun, Mexico, May 29-31, pp. 245 - 248.

Rodrigues L., Prado M., Tavares P., Da Silva K., Rosa A. (1996), “Simulation and control of biped locomotion-GA optimization”, Proc. of IEEE Int. Conf. on Evolutionary Computation, pp. 390 - 395.

Winston, P. (1992). “Artificial intelligence”, Reading MA: Addison-Wesley, pp. 505 - 528.

Zheng Y. F., Shen. J., Sias F. (1988). “A motion control scheme for a biped robot to climb sloping surfaces”, Proc. IEEE Int. Conf. on Robotic and Automation, Vol. 2, pp. 814 - 816.