INTELLIGENCE BY EVOLUTION

Prabhas Chongstitvatana

Department of Computer Engineering
Chulalongkorn University
Phayathai road, Bangkok 10330, Thailand
phone 02-2186982 fax 02-2186955
prabhas@chula.ac.th

ABSTRACT

In this talk, I will describe an alternative to achieve intelligent machines, using evolutionary computation. Robotics offers rich and fruitful area of research, it requires many cognition tasks and motion planning and control. Robots also work in the real world that requires interaction with human being. I will demonstrate three examples of my work in robotics that use evolutionary computation to create robot programs that perform tasks in the real world: visual-servo tasks, planning for a robot manipulator and a biped static walker.

KEYWORDS

Evolutionary computation, robot programming, genetic algorithms, genetic programming.

1. INTRODUCTION

Traditional approaches in Artificial Intelligence can be categorised into two schools: symbolic and subsymbolic. The symbolic AI relies on representing the domain of interest in some form of symbolic logic such as first order predicate calculus. Reasoning is performed on this representation using automate reasoning systems. Some examples of the success in this area are: expert system, symbolic mathematics (such as Mathematica), machine translation etc. The subsymbolic AI concentrates on analogical representation such as artificial neuron networks (ANN) or fuzzy logic that must be simulated or trained with sample data to achieve the required tasks. Some examples of the success in this area are: an automatic driver of a car using ANN, face recognition, adaptive control using fuzzy logic etc. In the domain of real world robots, both schools offer some advantage and some disadvantage. The symbolic AI is good at planning and reasoning tasks while the subsymbolic AI is good at sensing and cognition tasks. It is not clear what is the best way to combine their advantages.

One of the obvious capabilities that a machine should posses to enable it to become intelligent is the ability to learn. It is hoped that a learning machine can learn to improve itself. Presently machine learning has become the area of active research. Machine learning [1] tries to discover set of rules to distinguish between positive and negative examples and uses those rules to perform reasoning tasks on the new data. There are plenty of applications of machine learning techniques: data mining, knowledge discovery etc. Machine learning in a way, is bridging the gap between symbolic and subsymbolic AI.

Evolutionary computation is an attempt to use the knowledge inspired by natural evolution (in a broad sense) to understand the process of adaptation in artificial systems. The motivation is that as nature has succeeded in producing complex systems as good as human being, its guiding principle may be applicable to produce intelligent machines. Many subarea of researches are very active at present time: Genetic algorithms (GA) [2] and its variant, Genetic Programming (GP) [3], Classifier System (CFS) and Evolution Strategy (ES) [4]. These techniques can be summarised as using the search based on population and use non-linear, probabilistic operators to choose the next searching point. (This explanation over simplifies many aspects of evolutionary computation but it gives a simple picture of the field).

2 ROBOTICS BY EVOLUTIONARY COMPUTATION

Robots are highly engineered systems. There are many constraints in technology in building robots. In any complex engineering systems, it is very difficult create capabilities that are both effective and flexible in one system. For if a machine is to be efficient (and effective) it is usually doing one thing well. To give general capabilities to a robot, it must be programmable. Programming a machine that works in the real world full of uncertainty and changes is not easy for human to do. Perhaps "adaptation" is the key to achieve general capabilities. Evolutionary computation is very powerful technique for this purpose.

Genetic Programming (GP) which represents a solution by a tree structure is appropriate to represent robot programs. How a robot can be programmed using Genetic Programming will be explained. The general form of evolutionary computation is as follows:

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

Fig. 1 A general form of evolutionary computation

First, a number of individuals, called 'population', are initialised 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. In Genetic Programming (GP), an individual has the structure of a tree, which is suitable to represent a program. Each terminal node is a command to produce sensing or motions. Each internal node, or a connective, is some form of control flow command, such as if-then-else. The choice of commands is important to achieve a satisfactory result. This individual represents the program for a robot to achieve a specific task. Evaluation of these programs can be done both in simulation and by using the real robot to produce a 'fitness value' of each individual. Figure 2 shows a robot program. In this program the terminal set is { w+, w-, e+, e- …} which are motion commands (to move wrist, elbow joints) and {SEE? OUT? HIT? ….} are sensing commands. The function set is {IF-AND IF-OR IF-NOT}.

```
(IF-AND w+ w+ e+ (IF-AND (IF-NOT (IF-NOT OUT?
s- w+) (IF-AND e+ s- e- (IF-OR (IF-NOT w+ s+ e+) s+
e- e-)) (IF-OR (IF-NOT SEE? w- e-) w+ e+ e+)) w-
(IF-OR SEE? (IF-OR e- (IF-OR (IF-OR HIT? e+ s+ e-)
INC? w- e-) s+ w+) (IF-AND (IF-NOT w- e- e-) w- w-
w+) s-) e+)))
```

Fig. 2 A sample of robot program which is randomly generated.

3 THREE EXMAPLES OF ROBOTS THAT USE EVOLUTION

3.1 Planning for a manipulator [5,6,7]

The system consists of a robot arm and a camera. The camera looks at the arm and the environment (as in figure 3). The task is to move the hand to the target while avoiding obstacles.

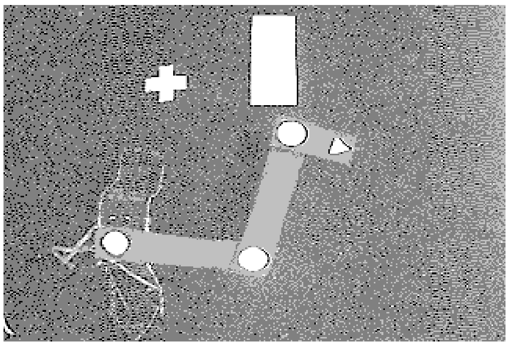

Fig. 3 The picture of the robot arm and the environment looking through the camera.

The target, the obstacles and the arm are represented directly in the image. Therefore checking the collision and the out of bound condition are done in the image plane. This is an analogical representation of the world.

In running the experiments, a forward kinematics model is built by instantly various parameters measured by the camera in the beginning of the run. This model is used during the simulated run of each program that is genetically generated. The set of terminals are identified to include six primitive servo motor functions and the system checking functions in the terminal set. Thus, the terminal set T for this problem is T = { s+, s-, e+, e-, w+, w-, HIT?, SEE?, INC?, DEC?, OUT?}. A set of functions for this problem, the function set F consists of F = { IF-AND, IF-OR, IF-NOT}.

The fitness of an individual is evaluated with the following function:

$$f = k_1 \times final + k_2 \times travel + k_3 \times opening \quad (1)$$

where *final* is the normalised distance between final position of the arm and the target, *travel* is a total normalised distance that the arm travelled, *opening* is a boolean value signified that the arm has reached an opening in the environment. A penalty value is applied when the arm hits the obstacles too many times. k_1, k_2, k_3 are weight factors. The fitness function is a cost function, the lower number means the better performance of the program. This function indicates that 1) getting close to the target lower the cost 2) move economically has the lower cost 3) moving to an opening to the target has the lower cost and 4) penalise the badly performed program by giving it a high cost. Figure 4 shows a sample of the result.

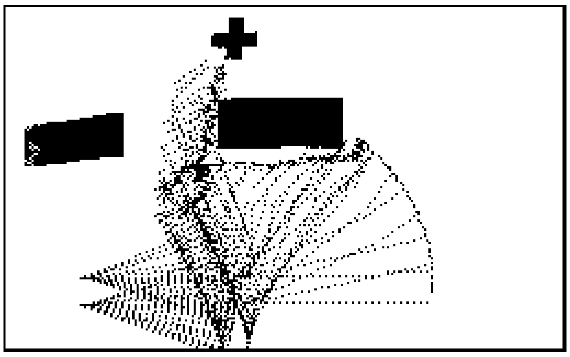

Fig. 4 A sample of experiment which show the movement of the manipulator

3.2 Visual servoing [8]

The control of a robotic hand-eye system requires the knowledge of the arm configuration and the camera position. The kinematic equations describing the system require precise information. Traditionally, the camera calibration is very important. However, in practice where the task requires the mobility of camera, retaining the calibration is very difficult. Numerous works have proposed the uncalibrated system for the control of robotic hand-eye system, which is called *visual servoing*. Some visual features are used as inputs to the control system. The image Jacobian is used to control the robot arm.

The estimate of the initial image Jacobian (J) is used to derive the control commands with the following relation [9]:

$$\Delta f = J\,\Delta q \qquad (2)$$

Where Δf is the change of image features (the position of the robot's end effector) in an image coordinate and Δq is the change of joints in the robot arm. It is the fact that an image Jacobian is correct only within a small region near the point which it is estimated.

This work introduces an adaptive image Jacobian system using Evolution Strategies. Evolution Strategies (ES) is a technique that mimics a natural evolution. The mutation operator creates offspring by adding all components of a parent with normally distributed random numbers. We use a (1+1)-ES system with adaptive σ with $c = 0.82$. ES is used to search among variants of Jacobian to find a good estimate of the Jacobian at the current position. We compare the proposed method with a system using a fixed Jacobian. The result shows that the proposed method achieves higher trajectory accuracy than a fixed Jacobian method.

Once the robot arm moved away from its initial calibrated position, the Jacobian needed to be adjusted. We used ES to generate a number of variants of the current Jacobian and used the knowledge about the motion that the robot has just been carried out to select the best estimate among these variants. Assume we generate *n* variants of the current Jacobian. These variants are used to find a set of commands to move the robot joints.

$$\Delta q_n = J_n^{-1}\,(\,s \bullet \Delta f\,) \qquad (3)$$

We calculate the expected positions of this set of commands using the previous Jacobian. Let denote the previous Jacobian J_0.

$$\Delta f_n = J_0\,\Delta q_n \qquad (4)$$

Using these expected positions, we choose J_n which can best predict the motion along the target line now that the actual motion (using J_0) is already known. The best J_n is the one that the expected position is the nearest complement of the actual position, where the complement position is measured by the angle between the target line and the line of motion. In other words, the best J_n is the one that, if it is used previously, the motion will be closer to the target line. This J_n is used as the Jacobian for the current position.

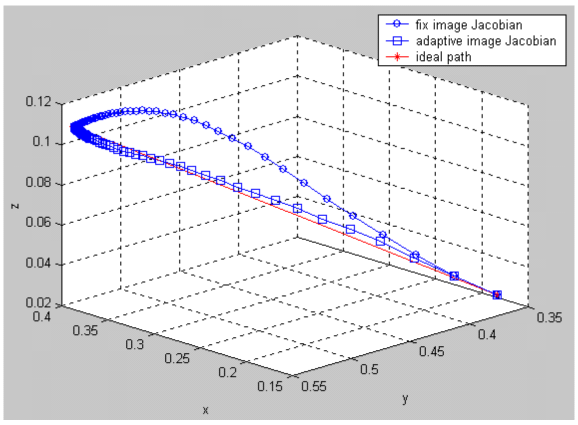

Fig. 5. Comparing the trajectory between Fixed and Adaptive Jacobian method.

The result shows that that the proposed method to estimate the image Jacobian using Evolution Strategies performed better than a fixed Jacobian system for a visual servoing task. The experiment clearly demonstrates that an adaptive Jacobian system using evolutionary approach can work very well. The main advantage of ES system is that it is possible to cope with high-dimensional problems.

3.3 Biped static walker [10,11]

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 [12]. 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 experimental biped is 25 centimeter high, and the area of sole is 4.5 * 5.0 cm^2. It has two hips, two knees, and two ankles, rotated in sagital plane (Fig.

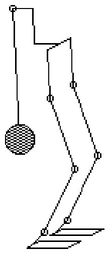

Fig. 6. Biped construction

6). The biped does not have a torso, but it has a tail moving in frontal plane. The reason of using tail instead of torso is that the tail will cause the biped 's center of gravity (C.G.) positioning lower than torso, so the biped can keep its balance easier.

An individual is composed of two parts: a length, and a sequence of walking commands. 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’.

Walking motion of one step is divided in to six stages.:
1. stand on the right foot
2. lift up the left leg
3. lay down the left leg
4. stand on the left foot
5. lift up the right leg
6. lay down the right leg

GA is used to synthesize control program for each stage step-by-step, called “stage evolution”. 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.

The initial biped posture is standing on two feet. In the first stage, robot stands on a right foot. In the second stage, lifting the left leg and laying down in the third stage. In the fourth stage, robot stands on the left foot. The fifth and sixth stage is lifting the right leg and laying it down. After the final stage, the posture is adjusted to the initial posture. Therefore, the control program can be repeated to create a continuous walking.

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

$$Fit = k_1 F + k_2 R / k_3 T \quad (5)$$

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 forward without turning.

The particular fitness function for each stage is shown in Table. 1. Both fitness functions are minimized function.

St	Fitness function
1	distance between C.G. and the center of right foot.
2	distance of the left foot forward from the right foot.
3	height of the left foot from the ground.
4	distance between C.G. and the center of left foot.
5	distance of the right foot forward to the left foot.
6	height of the right foot from the ground.

Table. 1. Particular fitness function for each stage

The experiment with an actual robot takes a very long time. During the evolution, the robot must perform a very large number of trial and error movements. To avoid this limitation, we use a simulation. The experiment with the real robot is performed to validate the solutions at the end of each stage. Human observation is used to validate the behavior of the robot.

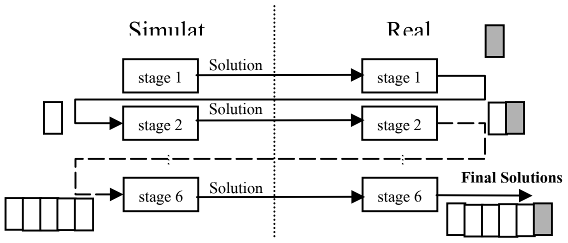

Fig. 7 Combination of simulated and real world experiments

At the end of each stage in simulation, 20 individuals are selected to be validated in the real experiment (Fig. 7). The validated result becomes an initial state of the next walking stage. After six stages, the total solutions will emerge.

The result shows that the robot can walk continuously more than 15 steps, with the speed 40 second per step. We found that even without general fitness function, the final solution could still be achieved. The cooperation between simulation and real world experiment is the key to achieve a solution that works in the real world.

4. CONSTRAST TO THE TRADITIONAL SOLUTIONS

The most well-known technique for path planning is the configuration space approach [13]. All geometry involving the environment and the manipulator must be known, the space is transformed into the configuration space where all possible free space in all possible configuration of the manipulator is represented. The search of path planning is done is this space. Extension of this approach to cope with uncertainty has been done, including sensing.

Visual-servoing has been a good example of feedback-type control application. However, it is not easy to incorporate both sensing and control in this task where the relationship between robot coordinate and camera coordinate can be changed continuously. The traditional attempt using the calculation of the relation between two coordinate becomes too complex to manage. The work on visual servoing has shown that there is some simpler relation that can be exploited in controlling the motion of robot arm, from the image itself. A work has been done in exploring this notion and achieved a very impressive result [14]. The work presented here is an alternative to solving a rank one updating formula in the Broyden formulas.

A biped walker was an ambition goal in robotic for a long time. A biped walker has stirred the interest of general public when Honda showed a series of humanoid robots that can walk [15]. A simple form of autonomous walking is static, i.e. the center of gravity is always within the supporting area, as opposed to dynamic walking which is what human and animals do. Walking is a repetitive motion hence is more amended to be analysed by control theory. Programming a biped to walk is equivalent to solving a form of dynamic equation of walking motion and find a controller to control all joint-motors.

5. CONCLUSION

I have demonstrated in three examples that evolutionary computation can be applied to program robots to achieve tasks in the real world. The main advantage of evolutionary approaches is that instead of using specialised techniques for each problem, one general technique can be used to solve all problems. Much implicit knowledge that is hard-coded into the solution in traditional approaches is made explicit in the evolutionary approaches in the form of evaluation function for fitness values. Performance is not impressive in this approach but its adaptation, the ability of make change to the system itself while performing the task, is the crucial property and the promise of this kind of system.

In the reference section you can find many introductory materials to the topic of this talk, many of them are standard textbooks. You can find more detailed description of all my work presented here in my research webpage, including a video clip of the biped walker [16]. Humanoid robots are becoming the active research topic worldwide and many research groups have produced exciting results. A specialised conference has been founded to discuss and publish the research results [17]. You can read about them in many research webpages available [18,19]. Evolutionary computation has been applied to wide range of problems including combinatorial optimisation and design automation. Its application in robotics is obvious but it is yet not clear how to find a scheme that can work in the real world because of intensive computation required. Yet, in my opinion, this is the most exciting aspect of achieving intelligent machines.

6. ACKNOWLEDGEMENT

Throughout my research in this topic I have been kindly funded by the following agencies, Thailand Research Fund, Faculty of Engineering Research Institure, Chulalongkorn university, National Science and Technology Development Agency and National Electronics and Computer Technology Centre. I would like to thank all my students, undergraduates and postgraduates whose diligent work have advanced the research in this area (the list of student names can be found in my research webpage [16]).

REFERENCES

[1] T. Mitchell, Machine Learning,, McGraw Hill, 1997.

[2] J. Holland, Adaptation in Natural and Artificial Systems. Ann Arbor, MI: University of Michigan Press, 1975.

[3] J. Koza, Genetic Programming. MIT Press, 1992.

[4] T. Back, F. Hoffmeister, H. Schwefel, "A survey of Evolution Strategies". In Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann, 1991.

[5] W. Suwannik and P. Chongstitvatana, "Improving the robustness of evolved robot arm control programs with multiple configurations", 2nd Asian Symposium on Industrial Automation and Robotics, Bangkok, Thailand, May 17-18, 2001, pp.87-90.

[6] P. Chongstitvatana and J. Polvichai, "Learning a visual task by genetic programming", Proc. of IEEE/RSJ Inter. Conf. on Intelligent Robots and Systems, Osaka, Japan, 1996.

[7] J. Polvichai and P. Chongstitvatana, "Visually-guided reaching by genetic programming", Proc. of 2nd Asian Conf. on Computer Vision, Singapore, 1995.

[8] K. Praditwong and P. Chongstitvatana, "An uncalibrated visual servoing using evolution strategies to estimate image Jacobian", 2nd Asian Symposium on Industrial Automation and Robotics, Bangkok, Thailand, May 17-18, 2001, pp.60-63.

[9] A. Conkie and P. Chongstitvatana, "An uncalibrated stereo visual servo system", Proc. of the British Machine Vision Conference, Oxford, 1990, pp.277-280.

[10] C. Chaisukkosol and P. Chongstitvatana, "Evolving control programs for a biped static walker", IEEE Inter. Conf. on Humanoid Robots, (poster paper), Waseda, Tokyo, November 22-24, 2001.

[11] C. Chaisukkosol and P. Chongstitvatana, "Automatic synthesis of robot programs for a biped static walker by evolutionary computation", 2nd Asian Symposium on Industrial Automation and Robotics, Bangkok, Thailand, May 17-18, 2001, pp.91-94.

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

[13] T. Lozano-Perez, J. Jones, E. Mazer and P. O'Donnell, HANDEY A Robot Task Planner. MIT Press, 1992.

[14] M. Jagersand, O. Fuentes and R. Nelson, "Experimental Evaluation of Uncalibrated Visual Servoing for Precision Manipulation". In Proc. of Inter. Conf. on Robotics and Automation, New Mexico, April 1997, pp.2874-2880.

[15] http://world.honda.com/robot/

[16] http://orange.cp.chula.ac.th/

[17] 2nd IEEE-RAS International Conference on Humanoid Robots November 22-24, 2001 Waseda University, Tokyo, Japan.

[18] http://www.humanoid.waseda.ac.jp/

[19] http://www.ai.mit.edu/projects/cog/